Automates infinis, logiques et langages

Arnaud Carayol

N° d'ordre: 3415

THÈSE

Présentée devant

devant l'Université de Rennes 1

pour obtenir

le grade de : Docteur de l'Université de Rennes 1
Mention Informatique

par

Arnaud Carayol

Équipe d'accueil :	Groupe Galion (Irisa)
École Doctorale :	Matisse
Composante universitaire :	IFSIC

Titre de la thèse :

Automates infinis, logiques et langages

Soutenue le 8 décembre 2006 à 14h00 en salle des thèses devant la commission d'examen

M. :	Jean-Claude	Raoult	Président
MM. :	Jacques	Sakarovitch	Rapporteurs
	Wolfgang	Thomas	
MM. :	Didier	Caucal	Examinateurs
	Colin	Stirling	
	Igor	Walukiewicz	

Remerciements

Je tiens tout d'abord à remercier les membres du jury qui m'ont fait l'honneur de prendre de leur temps pour évaluer ce mémoire.

Merci à Jacques Sakarovitch d'avoir évalué ce document dans un temps si restreint. Je le remercie pour ses remarques et pour l'intérêt qu'il a porté à ce travail. Je m'excuse ici pour les affres que lui ont causé la lecture de cet âpre document.

Merci à Wolfgang Thomas d'avoir accepté d'être rapporteur cette thèse. Je le remercie aussi grandement pour toutes les opportunités qu'il m'a données durant ma thèse en permettant ma collaboration avec Stefan Wörhle et en m'accueillant dans son équipe pour mon post-doctorat.

Many thanks to Colin Stirling who introduced me 6 years ago to this wonderful research area and who suggested Didier Caucal as my PhD avdisor. He originated this adventure and I am proud that he sees its conclusion.

Je remercie Igor Walukiewicz dont les travaux sur la décidabilité de la logique monadique ont nourri une grande partie de ma thèse.

Je remercie enfin Jean-Claude Raoult d'avoir accepté d'être président de ce jury. Au cours des années, son accueil et sa disponibilité pour répondre à mes questions naïves ont beaucoup compté pour moi.

Je ne pourrai en quelques lignes traduire l'immense privilège et plaisir d'avoir eu Didier Caucal pour directeur de thèse. Sa passion et sa vision de la recherche sont pour moi un modèle. Il n'y aucune exagération à dire que: sans son constant soutien, sans toutes ses qualités humaines, cette thèse n'aurait pas vu le jour. Je le remercie pour tout cela et pour bien plus encore.

Ce travail de thèse a été le fruit de nombreuses rencontres et collaborations.

Je tiens à remercier Antoine Meyer pour notre fructueuse et très enrichissante collaboration. Ses grandes qualités scientifiques, son amitié et son soutien jusqu'aux derniers instants de cette thèse m'ont été précieux. J'espère avoir dans l'avenir le plaisir de continuer avec lui ces travaux et de concrétiser les projets nés durant ces années de thèse.

Je remercie Stefan Wörhle avec qui nous avons joint nos forces pour étudier les graphes des automates à piles d'ordre supérieur. Son accueil et sa générosité ont été incomparables.

Je remercie aussi Thomas Colcombet pour son enthousiasme et pour tout ce qu'il m'a appris durant ma thèse.

Last but not least, je remercie les amis qui ont toujours été là même quand les preuves me fuyaient. Pour les croissants et le café les matins de nuits blanches, pour l'oreille à l'autre bout des coups de fils obnubilés et pour avoir subi bien d'autres facéties, je remercie du fond du coeur: Alexandra, Corentin, Julien, Sébastien et Steven.

Je remercie mes parents qui ont subi les hauts et les bas de cette thèse et en ont relu toutes les épreuves.

Enfin, je remercie Claire sans qui cela n'aurait pas beaucoup de sens.

Table des matières

Introduction

Cette thèse s'inscrit dans l'étude *algorithmique* et *structurelle* des graphes infinis de présentation finie.

Automates infinis

Un automate infini est un graphe simple orienté et étiqueté [1] de présentation finie. Le terme *présentation finie* signifie que la structure de ce graphe est décrite par une quantité finie d'information. Cette propriété est cruciale dans le cadre de l'informatique théorique puisqu'elle rend ces graphes «accessibles» aux ordinateurs et donc au traitement automatique.

Il existe deux grandes classes de présentations finies d'un graphe: les présentations *internes* et *externes*.
Les représentations *internes*, qui sont les plus naturelles, fixent un nommage explicite des sommets du graphe. Les sommets sont, par exemple, des mots sur un alphabet fini ou encore des termes finis. Les arcs du graphe sont alors donnés par une machine finie acceptant des couples de sommets. Des exemples de telles machines sont les automates à pile et les machines de Turing. Dans les graphes ainsi définis, les sommets sont des configurations de cette machine et les arcs représentent les étapes de calcul de la machine.
Les représentations *externes* ne fournissent pas de nommage explicite des sommets mais une description de la structure du graphe. Un exemple de présentation externe consiste à décrire un graphe par une suite finie de transformations qui construit ce graphe en partant d'un graphe fini ou d'un graphe infini de présentation finie.

La majeure partie des recherches sur les automates infinis concerne des propriétés qui sont indépendantes de la présentation choisie. Nous parlerons de propriétés structurelles par opposition aux propriétés liées aux noms des sommets. Bien entendu, les propriétés algorithmiques de ces automates dépendent de la présentation adoptée. Deux grandes familles de propriétés structurelles vont gui-

1. ayant un ensemble dénombrable de sommets

der cette thèse: l'expressivité des logiques décidables sur les automates infinis considérés et les langages acceptés par ces automates.

Logique

Une logique est un langage permettant d'exprimer des propriétés structurelles du graphe. Elle est donnée par un ensemble de formules et une relation de satisfaction qui lie les graphes[2] et les formules. De nombreuses logiques ont été introduites avec des expressivités variées; nous en considérerons essentiellement deux: la logique au premier ordre (FO) et la logique au second ordre monadique (MSO). La logique au premier ordre permet de quantifier sur les sommets du graphe. Ainsi nous pouvons, par exemple, exprimer que tout sommet du graphe admet un successeur. Du point de vue de l'expressivité, la logique au premier ordre ne peut exprimer que des propriétés locales. En particulier, il est impossible d'exprimer l'existence d'un chemin entre deux sommets dans la logique au premier ordre. La logique au second ordre monadique est un enrichissement de la logique au premier ordre où l'on autorise à quantifier sur des ensembles de sommets. Grâce à cet ajout, elle peut exprimer des propriétés non-locales comme l'accessibilité et la confluence.

Une logique est décidable sur un automate infini s'il existe une procédure automatique prenant en entrée une formule de la logique et décidant si l'automate infini satisfait cette formule.

L'étude des automates infinis ayant une logique décidable est motivée par l'intérêt qu'ils présentent pour la *vérification* des systèmes informatiques ayant un ensemble de configurations de taille non bornée. L'automate infini est une abstraction d'un tel système et les formules logiques expriment des propriétés sur le comportement de ce système. Si la modélisation du problème est correcte alors le système a un comportement donné si et seulement si l'automate infini associé satisfait la formule exprimant ce comportement. Un compromis doit être trouvé entre la richesse de la structure de l'automate infini et l'expressivité de la logique. En effet une logique trop expressive ne sera décidable que sur des automates infinis de structure extrêmement pauvre et vice et versa.

Il est communément admis que la logique monadique réalise un compromis satisfaisant entre ces deux aspects. En effet d'une part, son expressivité est suffisante pour exprimer la plupart des propriétés que l'on souhaite vérifier sur le comportement des systèmes informatiques. D'autre part, elle est décidable sur des familles assez riches d'automates infinis. En pratique pour des raisons d'efficacité de la procédure de décision, on considère des logiques moins expressives que la logique monadique comme les logiques temporelles ou les logiques modales.

2. vus comme des structures relationnelles

Langages

À la fin des années 50, Chomsky a introduit, dans [Cho59], une hiérarchie de familles de langages. Cette hiérarchie, aujourd'hui appelée hiérarchie de Chomsky, a posé les bases de la théorie des langages formels. Elle est définie à base de grammaires formelles et contient quatre niveaux: les langages rationnels, algébriques, contextuels et récursivement énumérables. Ces quatre familles ont été originellement définies par des restrictions syntaxiques sur les grammaires les engendrant. Depuis elles ont donné lieu à une étude approfondie et en particulier a de nombreuses caractérisations alternatives. En particulier pour chacune de ces familles de langages, une famille d'accepteurs finis correspondante a été donnée: les automates finis, les automates à pile, les machines de Turing travaillant en espace linéaire et les machines de Turing.

Récemment, ces familles de langages ont été caractérisés par des familles d'automates infinis (voir par exemple [CK02, Tho01]). Pour voir les automates infinis comme des accepteurs de langages, il suffit de leur adjoindre un ensemble de sommets dit initiaux ainsi qu'un ensemble de sommets dit finaux. Le langage accepté par un automate infini ainsi enrichi est l'ensemble des mots étiquetant un chemin allant d'un sommet initial à un sommet final. Ce langage est appelé la *trace* de l'automate. Cette notion généralise parfaitement la notion d'automate fini sur un monoïde libre; ce qui justifie la dénomination d'automate infini. Cette approche a permis de proposer une hiérarchie de familles d'automates infinis acceptant la hiérarchie de Chomsky: les automates finis, les graphes préfixe-reconnaissables [Cau96], les graphes rationnels [MS01] et les graphes de transitions des machines de Turing [Cau03b].

Historique

L'étude des automates infinis a débuté avec les travaux de Muller et Schupp [MS85] sur les graphes enracinés des automates à pile (aussi appelés *context-free graphs* en anglais). Un tel graphe a pour sommets les configurations de l'automate à pile accessibles depuis la configuration initiale et ses arcs sont donnés par les transitions de l'automate. Muller et Schupp établissent une propriété structurelle fondamentale de ces graphes. Pour cela, ils considèrent la suite des graphes obtenus en supprimant un sommet donné, puis les sommets à distance au plus 1 de ce sommet, puis les sommets à distance au plus 2, etc. Ils établissent que pour les graphes enracinés des automates à pile, cette suite ne contient qu'un nombre fini de composantes connexes non isomorphes. Cette propriété leur permet de montrer que ces automates infinis ont tous une théorie au second ordre monadique (MSO) décidable. Ils se ramènent à la décidabilité de MSO sur l'arbre binaire complet, établie par Rabin dans [Rab69].

Dans [Cou90], Courcelle élargit cette propriété à la famille plus générale des graphes HR-équationnels. Ces automates infinis sont engendrés par les grammaires déterministes de graphes. Là encore, la décidabilité de la théorie au second ordre monadique est obtenue en se ramenant au théorème de Rabin.

Dans [Cau96], Caucal généralise encore cette famille en définissant la classe des graphes préfixe-reconnaissables. Il en donne une présentation externe en montrant que ces graphes sont obtenus en *dépliant* un graphe fini et en appliquant une transformation de graphes appelée *substitution rationnelle inverse*. Comme ces deux transformations préservent la décidabilité de la théorie monadique et que les graphes finis ont trivialement une théorie monadique, tous les graphes de cette famille ont donc une théorie monadique décidable.

Ces trois familles de graphes possèdent des caractéristiques similaires.
Tout d'abord, elles admettent une présentation interne basée sur les automates à pile. En particulier, Stirling établit dans [Sti00] que les graphes préfixe-reconnaissables sont isomorphes aux graphes des transitions des automates à pile. Les graphes de transitions sont obtenus en «masquant» les ε-transitions dans les graphes enracinés des automates à pile. Pour prolonger le parallèle entre automates finis et automates infinis, les graphes des transitions sont obtenus par ε-fermeture des graphes enracinés. Une conséquence immédiate de ce résultat est que les traces de ces trois familles d'automates infini entre un unique sommet initial et un unique sommet final sont les langages algébriques.
Ensuite pour ces trois familles, la décidabilité de leur théorie au second ordre monadique peut se réduire à la décidabilité de l'arbre binaire complet. Plus précisément, tous ces graphes peuvent être obtenus par *interprétation monadique* dans l'arbre binaire complet. Dans [Bar97], Barthelmann établit que les graphes préfixe-reconnaissables sont d'une certaine manière maximaux pour cette propriété: tout graphe interprétable dans l'arbre binaire complet est isomorphe à un graphe préfixe-reconnaissable.

En 2002, deux pistes pour définir des familles d'automates infinis ayant une théorie au second ordre monadique décidable ont été proposées. La première, proposée dans [KNU02], consiste à considérer les graphes aussociés à des enrichissements des automates à pile que sont les automates à pile d'ordre supérieur. La seconde proposée dans [Cau02], consiste à généraliser la caractérisation externe des graphes préfixe-reconnaissables et à considérer les graphes obtenus en itérant le dépliage et la substitution rationnelle inverse en partant des graphes finis. La première est basée sur une présentation interne et la seconde sur un présentation externe.

Approche interne. Les automates à pile d'ordre supérieur ont été introduits dans les années 70 comme une généralisation des automates à pile [Aho69, Gre70, Mas76]. Alors qu'un automate à pile travaille sur une pile (de niveau 1) de sym-

boles, un automate à pile de niveau 2 travaille sur une pile contenant des piles de niveau 1 et non plus des symboles comme au niveau 1. Un automate à pile de niveau 2 peut (comme au niveau 1) empiler ou dépiler un symbole sur la plus haute pile de niveau 1. Les nouvelles opérations introduites au niveau 2 sont la copie de la plus haute pile de niveau 1 et sa destruction. Les automates à pile de niveau k sont définis de manière similaire. Ces automates ont dans les années 80 été étudiés comme des accepteurs de langages [Dam82, Eng83].

En 2002, Knapik, Niwinski et Urzyczyn ont, dans le cadre de l'étude des schémas récursifs d'ordre supérieur, défini des arbres infinis associés aux automates à pile d'ordre supérieur et ils ont montré que ces arbres infinis ont tous une théorie monadique décidable [KNU02].

Approche externe. Dans [Cau98, Cau02], Caucal définit une hiérarchie éponyme de familles d'automates infinis ayant une théorie monadique décidable. Le niveau 0 de cette hiérarchie contient la classe des graphes finis. Les graphes du niveau $n+1$ sont les graphes isomorphes aux graphes obtenus en appliquant un substitution rationnelle inverse au dépliage d'un graphe de niveau n. Comme nous l'avons déjà mentionné, tous les graphes de cette hiérarchie ont un théorie au second ordre monadique décidable. Le niveau 1 contient la classe des graphes isomorphes à un graphe préfixe-reconnaissable.

Ce document s'intéresse à l'étude de ces deux approches et à leurs liens.

Contributions

En collaboration avec Stefan Wöhrle dans [CW03], nous avons établi l'égalité entre ces deux approches. Nous avons montré que les graphes du n niveau de la hiérarchie introduite par Caucal sont, à isomorphisme près, les graphes des transitions des automates à pile de niveau k. Nous établissons aussi que la hiérarchie obtenue en itérant dépliage et substitutions rationnelles inverses coïncide avec la hiérarchie obtenue en itérant des opérations plus générales que sont l'opération l'itération de structure [Wal96a] et les transductions monadiques [Cou94]. Cette équivalence est une conséquence de résultats de commutation partielle pour diverses transformations de graphes préservant la décidabilité de MSO obtenus en collaboration avec Thomas Colcombet dans [CC03].

Ce résultat établit la robustesse et la pertinence de ces familles d'automates infinis. En effet ces deux transformations (l'itération de structure et la transduction monadique) sont à notre connaissance les transformations de graphes les plus générales préservant la décidabilité de la logique MSO.

Pour réaliser une étude détaillée des graphes associés aux automates à pile de niveau k, il est nécessaire d'avoir une représentation finie des ensembles de

piles de niveau k associés aux automates à pile de niveau k. En particulier, il faut une notion permettant de capturer l'ensemble de piles apparaissant dans une configuration accessible depuis une configuration donnée. Pour les automates à pile, un résultat fondamental est que l'ensemble des contenus de pile accessibles depuis une configuration donnée forme un sous-ensemble rationnel des piles vues comme des mots sur l'alphabet de pile. Dans [Car05], nous définissons une notion d'ensemble rationnel de piles de niveau k pour lesquels cette propriété est aussi vérifiée. Nous effectuons une étude détaillée des propriétés algébriques et algorithmiques de ces ensembles. En particulier nous donnons deux notions complémentaires d'accepteurs déterministes et complets pour ces ensembles et ainsi que des procédures de déterminisation de complexité minimales.

Nous fournissons aussi de nombreux arguments qui établissent l'aspect canonique de cette notion de rationalité pour les piles de niveau k. En particulier, nous montrons que ces ensembles correspondent aux ensembles définissables en logique sur la structure canonique associée aux piles de niveau k.

En exploitant les liens naturels entre les automates à pile de niveau k et les ensembles rationnels de piles de niveau k, nous pouvons donner des preuves plus élémentaires des résultats obtenus dans [CW03]. De plus, nous pouvons définir une famille de graphes associée sans ε-fermeture structurelle plus pertinente que les graphes enracinés : les graphes des configurations. Ces graphes sont définis par une restriction à un ensemble rationnel de configurations au lieu d'une restriction par accessibilité. Ils sont caractérisés de manière externe en remplaçant les substitutions rationnelles inverses par des substitutions finies inverses. Nous généralisons aussi la notion de graphe *préfixe-reconnaissable* à tout niveau et établissons que les graphes préfixe-reconnaissables de niveau k sont (à isomorphisme près) les graphes des transitions des automates à pile de niveau k.

En résumé, ces travaux généralisent à tous niveaux la plupart des résultats obtenus au niveau 1 à l'exception notable de la caractérisation géométrique de Muller et Schupp.

Durant cette thèse, nous avons en collaboration avec Antoine Meyer mené une étude systématique des accepteurs infinis pour les langages contextuels. Ces travaux ont été publiés dans [CM05, CM06a, CM06b] et ne seront pas présentés en détail dans ce document. Cependant une brève synthèse en est donnée dans le chapitre 2.

Plan

Le document est structuré comme suit.

Dans le chapitre 1, nous définissons les notions de bases utilisées dans ce document. En particulier, nous présentons la hiérarchie de Chomsky et la logique

monadique avec sa caractérisation par automates d'arbres à parité sur les arbres déterministes.

Dans le chapitre 2, nous présentons un survol des principales familles d'automates infinis. Ce chapitre sera en particulier l'occasion d'une présentation plus détaillée des résultats obtenus en collaboration avec Antoine Meyer autour des accepteurs infinis pour les langages contextuels.

Dans le chapitre 3, nous présentons les différentes transformations de graphes préservant la décidabilité de la théorie au second ordre monadique. Nous établissons plusieurs résultats liant les compositions de ces différentes transformations.

Dans le chapitre 4, nous présentons la notion de rationalité associée aux automates à pile d'ordre supérieur et nous étudions ses propriétés algébriques, algorithmiques et logiques.

Dans le chapitre 5, nous présentons les différentes familles de graphes associées aux automates à pile d'ordre supérieurs et nous en donnons différentes caractérisations internes et externes qui généralisent le cas des automates à pile.

Chapitre 1

Notions préliminaires

Dans ce chapitre, nous introduisons quelques notions et notations préliminaires. Le paragraphe 1.1 couvre les objets mathématiques de base. Le paragraphe 1.2 donne quelques éléments de la théorie des langages formels et présente la hiérarchie de Chomsky. Le paragraphe 1.3 présente les notions relatives aux graphes et aux arbres. Le paragraphe 1.4 présente les logiques usuelles que sont la logique du premier ordre (FO) et la logique du second ordre monadique (MSO). Nous rappelons en particulier la caractérisation par automates d'arbres avec conditions de parité de MSO sur les arbres déterministes.

1.1 Notions de base

Pour tout ensemble P, nous noterons 2^P l'ensemble des parties de P. Si P est un ensemble fini, nous noterons $|P|$ son cardinal. Si P est infini, nous noterons simplement $|P| = \infty$. Un ensemble est dénombrable s'il est en bijection avec l'ensemble $\mathbf{N}$ des entiers.

1.1.1 Mots

Pour tout ensemble Σ, nous noterons Σ^* l'ensemble des suites finies d'éléments de Σ. Si Σ est fini, nous parlerons aussi de *mots* sur Σ. Pour tout $w \in \Sigma^*$, $|w|$ désigne la longueur de la suite w et pour tout $k \in [1,|w|]$, $w(k)$ désigne le $k^{\text{ième}}$ symbole de la suite w. L'unique suite vide de Σ^* sera notée ε. Pour deux suites u et v dans Σ^*, nous noterons $u \cdot v$ la concaténation des deux suites.
Pour tous mots u et $v \in \Sigma^*$, u est un préfixe de v (noté $u \sqsubseteq v$) s'il existe un mot $w \in \Sigma^*$ tel que $v = u \cdot w$. Si de plus u est différent de v alors nous dirons que u est un préfixe strict de v et nous écrirons $u \sqsubset v$. Un ensemble $P \subseteq \Sigma^*$ est *clos par préfixe* si pour tout $v \in P$ et pour tout $u \in \Sigma^*$, $u \sqsubseteq v$ implique $u \in P$. Le plus long préfixe commun de P est le plus long mot de l'ensemble $\{u \mid \forall v \in P, u \sqsubseteq v\}$.

1.1.2 Relations

Une *relation* R sur un ensemble P est un sous-ensemble de $P \times P$. Pour un couple $c = (p,q) \in P \times P$, nous prenons $\pi_1(c) = p$ et $\pi_2(c) = q$. La relation R est *symétrique* si pour tous u et $v \in P$, $(u,v) \in R$ implique $(v,u) \in R$. La relation est *réflexive* si pour tout $u \in P$, $(u,u) \in R$. La relation R est *transitive* s'il existe u,v et $w \in P$ tels que $(u,v) \in R$ et $(v,w) \in R$ alors $(u,w) \in R$. Une *relation d'équivalence* R est une relation réflexive, symétrique et transitive. La *classe d'équivalence* d'un élément $p \in P$, notée $[p]_R$, est l'ensemble de tous les éléments de P en relation par R avec p (*i.e.* $[p]_R = \{q \in P \mid (p,q) \in R\}$). Deux classes d'équivalences de R sont soit égales, soit disjointes.

1.1.3 Fonctions et fonctions partielles

L'ensemble des fonctions partielles d'un ensemble P dans un ensemble Q sera noté $P \dashrightarrow Q$. Pour toute fonction $f \in P \dashrightarrow Q$, nous noterons $\mathrm{Dom}(f)$ (resp. $\mathrm{Im}(f)$) le domaine de f (resp. l'image de f). Comme pour les relations, nous noterons $f \circ g$ la composée de la fonction partielle f puis de la fonction g. Une fonction f (ou application) de P dans Q est une fonction partielle $f \in P \dashrightarrow Q$ telle que $\mathrm{Dom}(f) = P$. Nous étendons canoniquement toute fonction partielle de P dans Q en une fonction de 2^P dans 2^Q. Pour tout ensemble $R \subseteq P$, nous noterons $f^{-1}(P)$ l'ensemble des antécédents de P par f (*i.e.* $f^{-1}(P) = \{q \in \mathrm{Dom}(f) \mid f(q) \in P\}$)
Pour toute fonction partielle f injective de P dans Q, la fonction partielle inverse notée f^{-1} de Q dans P est définie pour tout $q \in Q$ par:

$$f^{-1}(q) = \begin{cases} p & \text{s'il existe } p \in P,\ f(p) = q, \\ \text{non définie} & \text{sinon.} \end{cases}$$

1.1.4 Monoïdes

Un monoïde $\mathcal{M}$ est donné par un couple $(M,\cdot)$ où M est un ensemble et où $\cdot$ est un produit interne sur M associatif et admettant un élément neutre noté $1_{\mathcal{M}}$. L'opération de produit est étendue aux sous-ensembles de M en prenant pour tous sous-ensembles P et Q de M, $P \cdot Q$ égal à $\{p \cdot q \mid p \in P \text{ et } q \in Q\}$. Nous définissons l'étoile de $P \subseteq M$, notée P^*, comme $\bigcup_{i \in \mathbb{N}} P^i$ où $P^0 = \{\varepsilon\}$ et où pour tout $i > 0$, $P^{i+1} = P \cdot P^i$. De plus, nous noterons $P^+ = \bigcup_{i>0} P^i$.

Un monoïde est engendré par $P \subseteq M$ si $M = P^*$ et nous dirons qu'il est finiment engendré s'il est engendré par un ensemble fini. De même, nous dirons que le monoïde est librement engendré par P si tout élément $m \in M$ s'écrit de manière unique comme $p_1 \cdots p_n$ pour $n \geq 0$ avec $p_1, \ldots, p_n$ dans P.

Un *morphisme* φ d'un monoïde $\mathcal{M} = (M,\cdot_{\mathcal{M}})$ dans un monoïde $\mathcal{N} = (N,\cdot_{\mathcal{N}})$ est une fonction de M dans N telle que $\varphi(1_{\mathcal{M}}) = \varphi(1_{\mathcal{N}})$ et pour tout p et $q \in M$, $\varphi(p \cdot_{\mathcal{M}} q) = \varphi(p) \cdot_{\mathcal{N}} \varphi(q)$. Un *isomorphisme* est un morphisme qui de plus est une bijection.

L'ensemble des mots sur Σ^* est un monoïde pour la concaténation dont l'élément neutre est le mot vide ε. Ce monoïde est librement engendré par Σ.

L'ensemble $\Sigma^* \times \Sigma^*$ peut être muni d'une structure de monoïde en considérant la concaténation composante par composante. L'élément neutre est $(\varepsilon,\varepsilon)$.

Si l'on considère un alphabet Σ réduit à un singleton, le monoïde $\Sigma^* \times \Sigma^*$ est isomorphe au monoïde $(\mathbf{N}^2,+)$ où $+$ désigne l'addition composante par composante et dont l'élément neutre est $(0,0)$. Ce monoïde est finiment engendré par $\{ (0,1),(1,0) \}$. Il n'est cependant pas librement engendré car $(1,1)$ est égal à $(1,0) + (0,1)$ et $(0,1) + (1,0)$.

1.1.5 Parties rationnelles

L'ensemble des parties rationnelles $\mathrm{Rat}(\mathcal{M})$ d'un monoïde $\mathcal{M} = (M,\cdot)$ est le plus petit ensemble de parties de M contenant les parties finies et fermé par produit, union et étoile.

Exemple 1.1.1. Considérons le monoïde libre sur $\Sigma = \{ a,b \}$. L'ensemble des mots contenant le facteur ab est rationnel et égal à $\{ a,b \}^* ab \{ a,b \}^*$. Son complémentaire est lui aussi un ensemble rationnel égal à b^*a^*.

Les parties rationnelles du monoïde produit $\Sigma^* \times \Sigma^*$ sont appelées les relations rationnelles.

1.1.6 Automates finis

Les automates finis sur un monoïde $\mathcal{M}$ fournissent une représentation finie des ensembles de $\mathrm{Rat}(\mathcal{M})$. Pour une présentation détaillée de cette notion, nous renvoyons le lecteur à par exemple [HU79, Sak03].

Un automate fini A étiqueté par un ensemble fini $P \subseteq M$ est donné par un quadruplet (Q,I,F,Δ) où Q est l'ensemble fini des états, I et F sont des sous-ensembles de Q correspondant respectivement aux états initiaux et finaux, et l'ensemble $\Delta \subseteq Q \times P \times Q$ est l'ensemble des transitions. Un calcul de l'automate A est une suite de $p_0 a_1 p_1 \dots a_n p_n \in Q(PQ)^*$ telle que pour tout $i \in [0,n-1]$, la transition (p_i,a_i,p_{i+1}) appartienne à Δ et telle que p_0 soit un état initial et p_n un état final; ce calcul accepte l'élément $a_0 \cdots a_n$ de M. L'ensemble de tous les éléments de M acceptés par A sera noté $\mathcal{L}(A)$. Les parties rationnelles de $\mathcal{M}$ sont les parties acceptées par un automate fini étiqueté par un sous-ensemble fini de M.

Un automate $A = (Q,I,F,\Delta)$ étiqueté par P est déterministe si pour tout $p,q,q' \in Q$, si (p,a,q) et (p,a,q') appartiennent à Δ alors $q = q'$. L'automate est complet si pour tout p et pour tout $a \in P$, il existe un état $q \in Q$ tel que $(p,a,q) \in \Delta$.

Exemple 1.1.2. Reprenons le langage rationnel $\{a,b\}^*ab\{a,b\}^*$ présenté dans l'exemple 1.1.1. La figure 1.1 donne deux automates acceptant ce langage. Celui de gauche n'est ni déterministe, ni complet et celui de droite est déterministe et complet. Les états initiaux sont marqués par une flèche entrante et les états finaux par une flèche sortante.

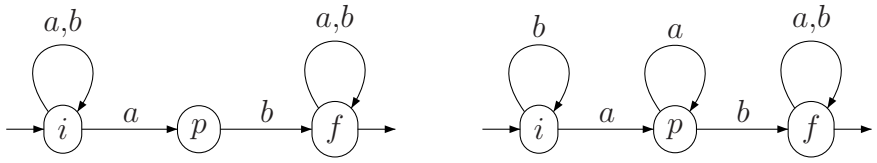


FIG. 1.1 – *Automates acceptant* $\{a,b\}^*ab\{a,b\}^*$.

1.1.7 Parties reconnaissables

Une partie $P \subseteq M$ d'un monoïde $\mathcal{M} = (M,\cdot)$ est reconnaissable s'il existe un morphisme φ de $\mathcal{M}$ à un monoïde fini tel que $P = \varphi^{-1}(\varphi(P))$. L'ensemble des parties reconnaissables de $\mathcal{M}$ est noté $\mathrm{Rec}(\mathcal{M})$. L'ensemble $\mathrm{Rec}(\mathcal{M})$ est une algèbre de Boole.

Dans le cas du monoïde libre Σ^* sur un ensemble fini Σ, les parties rationnelles et reconnaissables sont identiques [Kle56]. Pour un monoïde finiment engendré, les parties reconnaissables sont des parties rationnelles. La réciproque n'est en général pas vraie. Par exemple dans le cas du monoïde $(\mathbf{N}^2,+)$, les ensembles reconnaissables sont des unions finies d'ensembles de la forme $P \times Q$ où P et Q appartiennent à $\mathrm{Rat}((\mathbf{N},+))$ (cf. [Kni64]). L'ensemble $\{(n,n) \mid n \geq 0\} = (1,1)^*$ de $\mathrm{Rat}(\mathbf{N}^2,+)$ n'est donc pas un ensemble reconnaissable.

1.2 Hiérarchie de Chomsky

Dans ce sous-paragraphe, nous présentons la plus connue des hiérarchies de la théorie des langages formels qui a été introduite par Chomsky dans [Cho59]. Pour une présentation détaillée de ces familles de langages, nous renvoyons le lecteur à par exemple [HU79].

Dans l'approche de Chomsky, les langages sont définis par des grammaires. Une grammaire G est donnée par un uplet (N,Σ,S,P) où N est un ensemble fini de symboles non-terminaux, Σ est un ensemble fini de symboles terminaux, $S \in N$ est l'axiome de la grammaire et P est un sous-ensemble fini de $N^* \times (N \cup \Sigma)^*$. Un élément (u,v) de P est une production que nous noterons $u \to v$. La grammaire induit une relation de dérivation $\underset{G}{\longrightarrow}$ sur $(N \cup \Sigma)^*$ définie par:

$$\underset{G}{\longrightarrow} = \{(wuw',wvw') \mid w,w' \in (N \cup \Sigma)^* \text{ et } (u,v) \in P\}.$$

Un mot $u \in \Sigma^*$ est dérivé par G si $S \underset{G}{\longrightarrow}^* u$. Nous noterons $\mathcal{L}(G)$ l'ensemble des mots de Σ dérivés par G.

La hiérarchie de Chomsky contient quatre niveaux qui sont définis par des restrictions syntaxiques sur les grammaires.

- Sans restrictions sur la forme de la grammaire, les langages engendrés sont les langages *récursivement énumérables*.
- Les grammaires croissantes ou contextuelles (*i.e.* dont les productions sont de la forme (u,v) avec $|v| \geq |u|$ engendrent les langages *contextuels*.
- Les grammaires algébriques (*i.e.* dont les productions sont de la forme (A,v) avec $A \in N$) engendrent les langages *algébriques* ou *hors-contexte*.
- Les grammaires linéaires à droite (*i.e.* dont les productions sont de la forme (P,aQ) engendrent les langages *rationnels*.

Toutes ces familles de langages sont acceptées par des machines possédant un nombre fini d'états. Les langages récursivement énumérables sont acceptés par les machines de Turing. Les langages contextuels sont acceptés par les machines de Turing travaillant en espace linéaire, aussi appelées machines linéairement bornées (LBM). Les langages algébriques sont acceptés par les automates à pile et, comme nous l'avons vu dans le paragraphe précédent, les langages rationnels sont acceptés par les automates finis.

Nous donnons ici une définition de ces accepteurs comme des systèmes de transitions étiquetées. Pour une définition classique, nous renvoyons le lecteur à [HU79]. L'avantage de cette présentation est de mettre en lumière le lien entre ces accepteurs et les graphes infinis qu'ils induisent. La notion de système de transitions étiquetées est une reformulation de la notion de machine hors-ligne (voir par exemple [MS97]).

1.2.1 Systèmes de transitions étiquetées

Un système de transitions étiquetées (LTS) par Σ est donné par un ensemble de configurations $\mathcal{C}$ et par une famille de relations $(\overset{a}{\longrightarrow})_{a \in \Sigma \cup \{\tau\}}$ sur $\mathcal{C}$. Le symbole

τ est un symbole spécial n'appartenant pas à Σ qui correspond aux pas internes du système ou à ses actions inobservables. Traditionnellement dans le cas des accepteurs finis, les transitions étiquetées par τ sont étiquetées par ε et appelées des ε-transitions. Nous utilisons le symbole τ pour éviter toute confusion avec le mot vide ε.

Un calcul du LTS est une suite $c_0 a_1 c_1 \dots a_n c_n \in \mathcal{C}((\Sigma \cup \{\tau\})\mathcal{C})^*$ avec $n \geq 0$ et pour tout $i \in [0,n-1]$, $c_i \xrightarrow{a_i} c_{i+1}$. Ce calcul est étiqueté par $a_1,\dots,a_n$ dans $(\Sigma \cup \{\tau\})$. Pour tout $w \in (\Sigma \cup \{\tau\})^*$ et pour toutes configurations c et $c' \in \mathcal{C}$, nous écrirons $c \xrightarrow{w} c'$ s'il existe un calcul de c à c' étiqueté par w.

Si nous omettons les actions internes étiquetées par τ, nous écrirons, pour tout $w \in \Sigma^*$, $c \stackrel{w}{\Longrightarrow} c'$ s'il existe un calcul étiqueté par $w' \in (\Sigma \cup \{\tau\})^*$ tel que w soit le mot obtenu en effaçant les occurrences de τ de w'.

Un système de transitions étiquetées est déterministe si pour toutes configurations c_1,c_2 et c_3 et pour tout $x \in \Sigma \cup \{\tau\}$,

- $c_1 \xrightarrow{x} c_2$ et $c_1 \xrightarrow{x} c_3$ implique $c_2 = c_3$,
- $c_1 \xrightarrow{\tau} c_2$ et $c_1 \xrightarrow{x} c_3$ implique $x = \tau$ et donc $c_2 = c_3$.

Si l'on fixe un ensemble de configurations initiales $\mathcal{C}_I$ et un ensemble de configurations finales $\mathcal{C}_F$, un mot $w \in \Sigma^*$ est accepté par le système de transitions étiquetées si $c_i \stackrel{w}{\Longrightarrow} c_f$ pour une configuration initiale $c_i \in \mathcal{C}_I$ et $c_f \in \mathcal{C}_F$.

1.2.2 Machines de Turing

Une machine de Turing est intuitivement composée d'une bande bi-infinie composée de cases contenant chacune un symbole dans Γ, d'une tête de lecture positionnée sur cette bande et d'un nombre fini d'états de contrôle. Les actions que peut effectuer la tête de lecture sont lire et écrire le contenu de la la case courante et se déplacer d'une case vers la droite ou vers la gauche. Nous donnons une définition adaptée de [Cau03b, Mey05] des machines de Turing.

Définition 1.2.1. Une machine de Turing est donnée par un uplet (Σ,Γ, [,],Q,q_0, F,δ), où Σ et Γ sont des ensembles finis de symboles d'*entrée* et de *bande*, [et] $\notin \Gamma$ sont des *délimiteurs de bande*, Q est un ensemble fini d'*états*, $q_0 \in Q$ est l'état initial, $F \subseteq Q$ est l'ensemble des états finaux et Δ est un ensemble fini de *règles de transition* de l'une des formes suivantes :

$$\begin{array}{lll} p[\xrightarrow{x} q[+ & pA \xrightarrow{x} qB\epsilon & p] \xrightarrow{x} q]- \\ p[\xrightarrow{x} q[& pA \xrightarrow{x} qB & p] \xrightarrow{x} q] \\ pA \xrightarrow{x} q & pA \xrightarrow{x} qBA & p] \xrightarrow{x} qB] \end{array}$$

avec p,$q \in Q$, A,$B \in \Gamma$, $\epsilon \in \{+,-\}$ et $x \in \Sigma \cup \{\tau\}$

Une configuration d'une machine de Turing M est un mot de la forme uqv avec $uv \in [\Gamma^*]$, $v \neq \varepsilon$ et $q \in Q$, où uv représente le contenu de la bande avec ses délimiteurs, q est l'état de contrôle courant et la tête de lecture pointe sur la cellule qui contient la première lettre de v. Nous noterons $\mathcal{C}_M$ l'ensemble des configurations de M. L'unique configuration initiale de M est $q_0[]$. L'ensemble des configurations finales de M est l'ensemble des configurations ayant un état final de M.

La machine de Turing M induit un système de transitions étiquetées défini par pour tout $x \in \Sigma \cup \{\tau\}$ par:

$$\begin{aligned}
\xrightarrow[M]{x} \;=\; & \{(upAv,uBqv) \in \mathcal{C}^2 \mid pA \xrightarrow{x} qB+ \in \Delta\} \\
& \cup \{(upAv,uqBv) \in \mathcal{C}^2 \mid pA \xrightarrow{x} qB \in \Delta\} \\
& \cup \{(uCpAv,uqCBv) \in \mathcal{C}^2 \mid pA \xrightarrow{x} qB- \in \Delta \text{ et } C \in \Gamma \cup \{[\,]\}\} \\
& \cup \{(upAv,uqv) \in \mathcal{C}^2 \mid pA \xrightarrow{x} q \in \Delta\} \\
& \cup \{(upAv,uqBAv) \in \mathcal{C}^2 \mid pA \xrightarrow{x} qBA \in \Delta\}.
\end{aligned}$$

Les machines de Turing déterministes (*i.e.*. qui induisent un système de transitions étiquetées déterministe) acceptent les même langages que les machines de Turing (non-déterministes). Ces langages sont fermés par union et intersection mais pas par complémentaire.

1.2.3 Machines de Turing linéairement bornées

La définition d'une machine de Turing linéairement bornée comme un système de transitions étiquetées est plus délicate. En effet, il faut assurer un lien linéaire entre la taille de la bande de la machine et le nombre de lettre de Σ qui ont été lues. La définition que nous présentons à été proposée dans [CM05].

Définition 1.2.2. Une machine linéairement bornée (LBM) est une machine de Turing $M = (\Sigma,\Gamma,[,],Q,q_0,F,\Delta)$ dont aucune règle d'insertion $pB \xrightarrow{x} qAB \in \Delta$ n'est étiquetée par τ.

On vérifie aisément que pour toutes configurations c et c' et pour tout $w \in \Sigma^*$ tel que $c \xRightarrow[M]{w} c'$ alors $|c'| - |c| \leq |w|$. En particulier, si l'on peut atteindre une configuration c depuis la configuration initiale en lisant un mot w (*i.e.* $q_0[] \xrightarrow{w} c$) alors $|c| \leq |w| + 3$: ce qui assure que la machine travaille bien en espace linéaire.

Comme nous l'avons déjà mentionné, les langages acceptés par les machines linéairement bornées sont les langages contextuels qui sont strictement inclus dans les langages récursivement énumérables. Ces langages forment une algèbre de Boole [Imm88]. Les machines linéairement bornées déterministes acceptent les langages contextuels déterministes. Contrairement aux cas des machines de

Turing, nous ne savons pas si les langages contextuels coïncident avec les langages contextuels déterministes [Kur64].

1.2.4 Automates à pile

Les automates à pile travaillent, comme leur nom l'indique, sur une pile. Ils peuvent lire le dernier symbole de cette pile, le dépiler ou empiler un nombre fini de symboles.

Définition 1.2.3. Un automate à pile P est donné par un uplet $(\Sigma,\Gamma,Q,q_0,I,\Delta)$ où Σ et Γ sont des ensembles finis de symboles d'*entrée* et de *pile*, Q est un ensemble fini d'*états*, $q_0 \in Q$ est l'état initial, $F \subseteq Q$ est l'ensemble des états finaux et où $\Delta \subseteq Q \times (\Gamma \cup \{\varepsilon\}) \times (\Sigma \cup \{\tau\}) \times Q \times \Gamma^*$ est l'ensemble des transitions.

Une configuration de P est un couple (p,w) où p est un état de Q et où $w \in \Gamma^*$ est une pile. L'unique configuration initiale de P est (q,ε). L'ensemble des configurations finales est $F \times \Gamma^*$. L'automate à pile P induit le système de transitions étiquetées défini pour tout $x \in \Sigma \cup \{\tau\}$ par:

$$\begin{array}{rcl} \xrightarrow[P]{x} & = & \{((p,wA),(q,wu)) \mid (p,A,x,q,u) \in \Delta\} \\ & & \cup\ \{((p,\varepsilon),(q,u)) \mid (p,\varepsilon,x,q,u) \in \Delta\}. \end{array}$$

Les langages acceptés par les automates à pile sont les langages algébriques qui sont strictement inclus dans les langages contextuels. Ces langages sont fermés par union mais ni par intersection, ni par complémentaire. Les langages algébriques déterministes (acceptés par les automates à pile déterministes) forment une sous-famille stricte des langages algébriques.

1.3 Graphes colorés

Dans tout ce document, nous fixons deux ensembles dénombrables Λ et Θ qui vont jouer le rôle de réservoirs pour les ensembles d'étiquettes et de couleurs de nos graphes.

Un graphe G orienté, étiqueté et coloré est un sous-ensemble de $V \times \Lambda \times V \cup \Theta \times V$ pour un certain ensemble dénombrable V. L'ensemble des étiquettes de G est $\Lambda_G := \{a \in \Lambda \mid \exists u,v \in V, (u,a,v) \in G\} \subseteq \Lambda$ et son ensemble de couleurs $\Theta_G := \{c \mid \exists u, (c,u) \in G\} \subseteq \Theta$. Nous supposerons toujours que ces deux ensembles sont finis. L'ensemble des sommets de G est le sous-ensemble V_G défini par:

$$\begin{array}{rcl} V_G & = & \{v \in V \mid \exists u \in V, \exists a \in \Lambda, (u,a,v) \in G \text{ ou } (v,a,u) \in G\}, \\ & \cup & \{v \in V \mid \exists c \in \Theta, (c,v) \in G\}. \end{array}$$

Pour tout u et v dans V_G et pour toute étiquette $a \in \Lambda$, si (u,a,v) appartient à G, nous dirons qu'il existe un arc étiqueté par a de u à v. Pour tous $u \in V_G$ et $c \in \Theta$, si $(c,u) \in G$, nous dirons que u est coloré par c. Pour simplifier notre présentation, nous supposerons par la suite que G ne contient pas de sommets isolés: tout sommet de V_G est soit source soit destination d'un arc de G.

Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$, un graphe G sur (Σ,C) ou (Σ,C)-graphe est un graphe tel que $\Lambda_G \subseteq \Sigma$ et $\Theta_G \subseteq C$. Si $C = \emptyset$, nous dirons que G est sans couleur et nous parlerons simplement de graphe sur Σ au lieu de graphe sur $(\Sigma,\emptyset)$.

Exemple 1.3.1. Nous définissons un graphe G sur (Σ,C) avec $\Sigma = \{\, a,b \,\}$ et $C = \{\, 1,2 \,\}$. Les sommets de G sont des mots sur l'alphabet $\Gamma = \{\, A,B \,\}$.

$$\begin{aligned} G \;=\; & \{(A^n,a,A^{n+1}) \mid n \geq 0\} \\ & \cup\; \{(A^nB^m,b,A^nB^{m+1}) \mid n \geq 0 \text{ et } m+1 \leq n\} \\ & \cup\; \{(1,A^n) \mid n \geq 0\} \cup \{(2,A^nB^n) \mid n \geq 0\}. \end{aligned}$$

L'ensemble V_G des sommets de G est égal à $\{A^nB^m \mid n \geq 0 \text{ et } m \leq n\}$. Le graphe G est présenté dans la figure 1.2. Les couleurs sont mises entre parenthèses à coté du sommet correspondant.

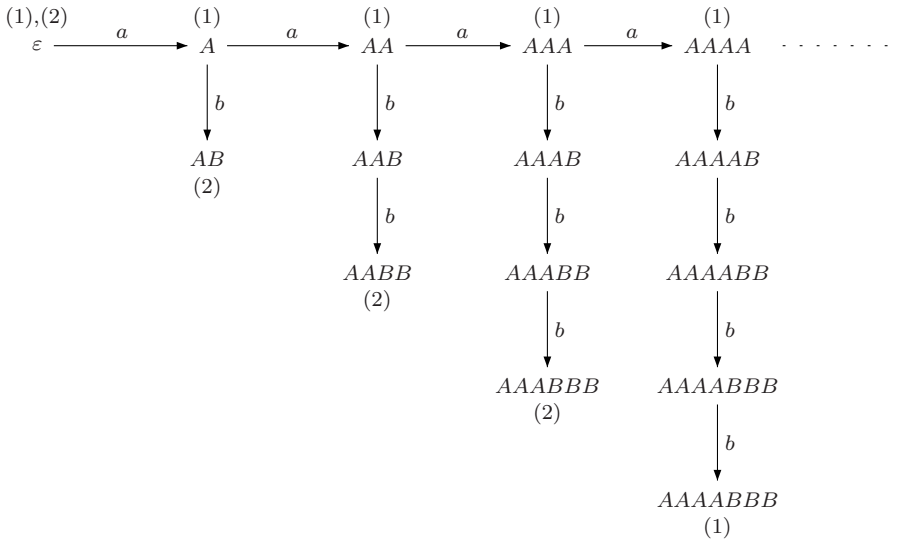


FIG. 1.2 – *Illustration d'un graphe.*

Un graphe G est *déterministe* si pour tout $a \in \Lambda$ et pour tout u,v et $v' \in V_G$, $(u,a,v) \in G$ et $(u,a,v') \in G$ implique que $v = v'$. Un graphe sur (Σ,C) est *complet* si pour tout $v \in V_G$ et pour tout $a \in \Sigma$, il existe $v \in V_G$ tel que $(u,a,v) \in G$. Pour tout $u \in V_G$, le *degré sortant* (resp. entrant) est $d^+(u) = |\{v \mid \exists a \in \Lambda, \exists v \in$

V_G,$(u,a,v) \in G\}|$ (resp. $d^-(u) = |\{v \mid \exists a \in \Lambda, \exists u \in V_G, (u,a,v) \in G\}|$). Le degré de u, noté $d(u)$, est la somme de ses degrés entrant et sortant.

Deux graphes G et H sont *isomorphes* (noté $G \approx H$) s'il existe une bijection h de V_G dans V_H telle que pour tout $a \in \Lambda$ et pour tout $c \in \Theta$ et $u,v \in V_G$, $(u,a,v) \in G$ si et seulement si $(h(u),a,h(v)) \in H$ et $(c,u) \in G$ si et seulement si $(c,h(u)) \in H$.

Un *chemin* dans un graphe G partant d'un sommet $u \in V_G$ et arrivant à un sommet $v \in V_G$ est une suite $v_0 a_1 v_1 \dots a_n v_n \in V_G(\Lambda_G V_G)^*$ pour $n \geq 0$ telle que $v_0 = u$, $v_n = v$ et telle que pour tout $i \in [1,n]$, (v_{i-1},a_i,v_i) appartienne à G. Dans la suite, nous noterons un tel chemin $v_0 \xrightarrow[G]{a_1} v_1 \dots v_{n-1} \xrightarrow[G]{a_n} v_n$. Nous dirons que ce chemin est étiqueté par $\varepsilon \in \Lambda_G^*$ si $n = 0$ et par $a_1 \cdots a_n \in \Lambda_G^*$ sinon. S'il existe un chemin étiqueté par $w \in \Lambda_G^*$ de u à v nous écrirons $u \xrightarrow[G]{w} v$ ou simplement $u \xrightarrow{w} v$ lorsque G découle du contexte.
Nous étendons cette notion pour prendre en compte les couleurs et les arcs pris dans le sens inverse. Pour indiquer qu'un arc est pris dans le sens inverse, nous considérons un ensemble $\overline{\Lambda}$ disjoint de Λ mais en bijection avec Λ. Pour tout $a \in \Lambda$, nous noterons $\bar{a}$ le symbole correspondant dans $\overline{\Lambda}$. Par analogie, pour tout sous-ensemble $\Sigma \subseteq \Lambda$, $\overline{\Sigma}$ désigne l'ensemble $\{\bar{a} \mid a \in \Sigma\}$.

Pour tout graphe G et pour tout $w \in (\Lambda \cup \overline{\Lambda} \cup \Theta)^*$, nous définissons par récurrence sur la longueur de w la relation $\xrightarrow[G]{w} \subseteq V_G \times V_G$ par:

$$\begin{array}{rclcrcl}
\xrightarrow[G]{\varepsilon} & = & \{(v,v) \mid v \in V_G\} & & \xrightarrow[G]{wc} & = & \{(u,v) \mid u \xrightarrow[G]{w} v \wedge (c,v) \in G\} \\
\multicolumn{3}{r}{\xrightarrow[G]{wa} = \{(u,v) \mid \exists u' \in V_G, u \xrightarrow[G]{w} u' \wedge (u',a,v) \in G\}} & & & & \\
\multicolumn{3}{r}{\xrightarrow[G]{w\bar{a}} = \{(u,v) \mid \exists u' \in V_G, u \xrightarrow[G]{w} u' \wedge (v,a,u') \in G\}.} & & & &
\end{array}$$

Intuitivement, un symbole $a \in \Lambda$ dans l'étiquette du chemin indique qu'un arc étiqueté par a est traversé, un symbole $\bar{a}$ indique qu'un arc étiqueté par a est traversé en sens inverse, et enfin une couleur $c \in \Theta$ indique que le sommet courant est coloré par c.

Exemple 1.3.2. Dans le graphe G présenté dans l'exemple 1.3.1, le chemin $A \xrightarrow[G]{a} AA \xrightarrow[G]{a} AAA \xrightarrow[G]{b} AAAB \xrightarrow[G]{b} AAABB$ part de A et arrive en $AAABB$. Nous avons donc $A \xrightarrow[G]{aabb} AAABB$. Il est aisé de vérifier que $AA \xrightarrow[G]{bb1\bar{b}\bar{b}2} AA$ alors que $AA \not\xrightarrow[G]{b1\bar{b}2} AA$.

1.3.1 Arbres

Un *arbre* T est un graphe tel qu'il existe un sommet $r \in V_T$, appelé la racine de T, tel que pour tout sommet $u \in V_T$, il existe un unique chemin dans T partant

de r et arrivant en u. À isomorphisme près, un arbre déterministe est entièrement caractérisé par une fonction partielle t de Λ_T^* dans 2^C telle que $\mathrm{Dom}(t)$ ne soit pas vide et soit clos par préfixe.
Plus précisément, à chaque arbre déterministe T de racine $r \in V_T$, nous associons la fonction partielle t_T de Λ_T^* dans 2^{Λ_T} définie pour tout $w \in \Lambda_T^*$ par:

$$t_T(w) = \begin{cases} \{c \mid (c,v) \in T\} & \text{s'il existe } v \in V_T,\ r \xrightarrow[T]{w} v \\ \text{non définie} & \text{sinon} \end{cases}$$

Il est aisé de vérifier que $\mathrm{Dom}(t_T)$ n'est pas vide et est clos par préfixe. Réciproquement pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$, nous associons à chaque fonction t de Σ^* dans 2^C de domaine non-vide et clos par préfixe, un arbre déterministe T_t sur (Σ,C) défini par:

$$T_t = \{(u,a,ua) \mid ua \in \mathrm{Dom}(t)\} \ \cup\ \{(c,u) \mid c \in t(u) \text{ et } u \in \mathrm{Dom}(t)\}.$$

Pour tout arbre déterministe T, T est isomorphe à T_{t_T}. Dans la suite, nous ne distinguerons pas la fonction de l'arbre qui lui est associé.
Les éléments de $\mathrm{Dom}(t)$ sont appelés les nœuds de l'arbre. Le nœud ε sera appelé la racine de l'arbre. Une branche infinie est une suite $(x_i)_{i\in\mathbf{N}}$ de nœuds de t telle que $x_0 = \varepsilon$ et pour tout $i \in \mathbf{N}$, $x_{i+1} = x_i a_i$ pour un certain $a_i \in \Sigma$. Pour tout nœud y de t, le sous-arbre t enraciné de y est $t_{/y}$ défini pour tout $w \in \Sigma^*$, par $t_{/y}(w) = t(yw)$.

Exemple 1.3.3. Le graphe G présenté dans l'exemple 1.3.1 est un (Σ,C)-arbre de racine ε. La fonction partielle t de Σ^* dans 2^C associée à G est définie pour tout $w \in \Sigma^*$ par:

$$t(w) = \begin{cases} \{1,2\} & \text{si } w = \varepsilon, \\ \{1\} & \text{si } w \in a^*, \\ \{2\} & \text{si } w = a^n b^n \text{ pour un certain } n \geq 1, \\ \emptyset & \text{si } w = a^n b^m \text{ pour } n \geq 1 \text{ et } 1 \leq m < n, \\ \text{non définie} & \text{sinon} \end{cases}$$

L'unique branche infinie de t est $(a^n)_{n\geq 0}$.

1.4 Logiques

Dans ce paragraphe, nous présentons la logique au premier ordre (FO) (cf. sous-paragraphe 1.4.2) et la logique au second ordre monadique (MSO) (cf. sous-paragraphe 1.4.3). Le sous-paragraphe 1.4.4 introduit les automates d'arbres avec conditions de parité qui caractérise MSO sur les arbres déterministes. Le sous-paragraphe 1.4.5 présente quelques résultats de base sur les jeux de parité et les

conséquences de ces résultats sur les automates d'arbres avec conditions de parité. Pour une présentation détaillée de ces notions, nous renvoyons le lecteur à [EF95] et [Tho97]. Enfin, le sous-paragraphe 1.4.6 présente la propriété de sélection pour MSO et quelques graphes possédant cette propriété.

1.4.1 Structures relationnelles

Une *signature* $\mathcal{S}$ est un ensemble fini de symboles possédant chacun une *arité*. Pour tout $R \in \mathcal{S}$, nous noterons $|R| \geq 1$ l'arité du symbole R. Une structure relationnelle $\mathcal{R}$ sur la signature $\mathcal{S}$ est donnée par un couple $(U,(R^{\mathcal{R}})_{R\in\mathcal{S}})$ où U est l'univers de $\mathcal{R}$ et où pour tout $R \in \mathcal{S}$, $R^{\mathcal{R}}$ est un sous-ensemble de $U^{|R|}$.

À tout graphe G, nous associons une structure relationnelle $\mathcal{G}$ sur la signature $\mathcal{S}_G$. La signature $\mathcal{S}_G$ est égale à $\{E_a \mid a \in \Lambda_G\} \cup \{P_c \mid c \in \Theta_G\}$ où pour tout $a \in \Lambda_G$, $|E_a| = 2$ et où pour tout $c \in \Theta_G$, $|P_c| = 1$. La structure $\mathcal{G}$ a pour univers V_G et pour tout $a \in \Lambda_G$, $E_a^{\mathcal{G}} = \{(u,v) \mid (u,a,v) \in G\}$ et pour tout $c \in \Theta$, $P_c^{\mathcal{G}} = \{u \mid (c,u) \in G\}$. Dans la suite, nous ne distinguerons pas G de la structure qui lui est associée.

1.4.2 Logique au premier ordre

Considérons un ensemble dénombrable $\mathcal{V}$ de variables du premier ordre. Nous utiliserons les lettres minuscules $x,y,z\ldots$ pour désigner ces variables du premier ordre. Les formules sur une signature $\mathcal{S}$ sont définies par récurrence. Les formules atomiques sont de la forme $x = y$ ou $R(x_1,\ldots,x_{|R|})$ où $x,y,x_1,\ldots,x_{|R|}$ sont des variables de $\mathcal{V}$ et où R est un symbole de $\mathcal{S}$. Si φ et ψ sont des formules sur $\mathcal{S}$ alors $\neg\varphi$ et $\varphi \wedge \psi$ sont aussi des formules sur $\mathcal{S}$. Enfin si ψ est une formule sur $\mathcal{S}$ alors pour toute variable $x \in \mathcal{V}$, $\exists x, \varphi$ est aussi une formule sur $\mathcal{S}$. L'ensemble des variables libres est défini de manière usuelle(cf. [EF95]). Nous écrirons $\varphi(x_1,\ldots,x_n)$ pour indiquer que les variables libres de φ appartiennent à l'ensemble $\{x_1,\ldots,x_n\}$. Une formule sans variables libres est aussi appelée un *énoncé* ou une *formule close*.

Nous noterons $\mathcal{R} \models \varphi$ le fait que la structure $\mathcal{R}$ sur la signature $\mathcal{S}$ satisfait l'énoncé φ sur $\mathcal{S}$. Cette notion est définie de manière usuelle. Pour une formule $\varphi(x_1,\ldots,x_n)$, et pour des éléments $u_1,\ldots,u_n$ de l'univers de $\mathcal{R}$, nous noterons $\mathcal{R} \models \varphi[u_1,\ldots,u_n]$ si $\mathcal{R}$ satisfait φ lorsque pour tout $i \in [1,n]$, la variable libre x_i est interprétée par l'élément u_i. Il n'est pas nécessaire que la signature de φ coïncide avec la signature de $\mathcal{R}$. En effet, il suffit d'adopter la convention que tous les symboles apparaissant dans la signature de φ mais n'apparaissant pas dans la signature $\mathcal{R}$ sont interprétés dans $\mathcal{R}$ par l'ensemble vide.

La théorie au premier ordre d'une structure $\mathcal{R}$ sur la signature $\mathcal{S}$ est l'ensemble des énoncés sur $\mathcal{S}$ satisfait par $\mathcal{R}$. Une structure $\mathcal{R}$ a une théorie au premier ordre

décidable si sa théorie au premier ordre est un ensemble récursif. Remarquons que la théorie d'un graphe est invariante par isomorphisme.

Nous utiliserons dans la suite la quantification universelle $\forall$, la disjonction $\vee$, l'implication $\rightarrow$, etc. Toutes ces constructions peuvent s'exprimer à l'aide de la négation $\neg$, la conjonction $\wedge$ et la quantification existentielle $\exists$. Cependant si elles n'enrichissent pas le pouvoir d'expression de la logique, elles permettent une écriture plus concise.

Exemple 1.4.1. Reprenons le graphe G présenté dans l'exemple 1.3.1. Ce graphe G satisfait l'énoncé $\varphi = \forall x, P_1(x) \rightarrow (\exists y, E_a(x,y))$ qui exprime que tous les sommets de G colorés par 1 sont la source d'un arc étiqueté par a. Considérons la formule $\psi(x,y) = \exists z, E_b(x,z) \wedge E_b(z,y) \wedge P_2(y)$. Pour tous u et $v \in V_G$, $G \models \psi[u,v]$ si et seulement si $u \xrightarrow[G]{bb2} v$.

Intuitivement la logique au premier ordre ne peut exprimer que des propriétés locales. Par exemple, il n'existe pas de formule $\varphi(x,y)$ exprimant l'existence d'un chemin entre x et y dans un graphe. Ce résultat peut être établi en utilisant le théorème de Gaifman qui donne un sens précis au caractère local de la logique du premier ordre. Nous renvoyons le lecteur à [EF95] pour une présentation détaillée.

1.4.3 Logique au second ordre monadique

La logique au second ordre monadique (MSO) étend la logique au premier ordre en ajoutant la possibilité de quantifier sur des ensembles d'éléments de l'univers de la structure.

Formellement, nous considérons deux ensembles dénombrables et disjoints de variables $\mathcal{V}_0$ et $\mathcal{V}_1$. L'ensemble $\mathcal{V}_0$ correspond aux variables du premier ordre que nous désignerons par des lettres minuscules $x,y,z,\ldots$. L'ensemble $\mathcal{V}_1$ correspond aux variables du second ordre monadique qui vont être interprétées par des ensembles d'éléments et que nous noterons par des lettres majuscules $X,Y,Z,\ldots$.

Les formules atomiques sont de la forme $x = y$, $x \in X$ ou $R(x_1,\ldots,x_{|R|})$ où $x,y,x_1,\ldots,x_{|R|}$ sont des variables de $\mathcal{V}_0$, $X \in \mathcal{V}_1$ et où R est un symbole de $\mathcal{S}$. Si φ et ψ sont des formules sur $\mathcal{S}$ alors $\neg\varphi$ et $\varphi \wedge \psi$ sont aussi des formules sur $\mathcal{S}$. Enfin si ψ est une formule sur $\mathcal{S}$ alors pour toute variable $x \in \mathcal{V}_0$ et $X \in \mathcal{V}_1$, $\exists x, \varphi$ et $\exists X, \varphi$ est aussi une formule sur $\mathcal{S}$. Les différentes notations et notions définies pour la logique au premier ordre sont adaptées à la logique au second ordre monadique.

Nous utiliserons les raccourcis de notations habituels tels que $\emptyset$, $X \cup Y$, $X \cap Y$, $X \subseteq Y$, etc qui peuvent tous être définis en MSO.

Pour tout graphe G, un sommet $u \in V_G$ est *définissable par MSO* s'il existe une formule $\varphi(x)$ telle que pour tout $v \in V_G$, $G \models \varphi[v]$ implique $u = v$. Cette

notion s'étend naturellement aux sous-ensembles de V_G.

Exemple 1.4.2. Nous pouvons exprimer en logique monadique l'existence d'un chemin reliant deux sommets. Fixons la signature $\{ E_a,E_b \}$ correspondant aux graphes étiquetés par $\Sigma = \{ a,b \}$. Considérons maintenant la formule $\varphi(x,y)$ donnée par:

$$\begin{array}{lcl} \varphi(x,y) & = & \exists Y, (\forall Z, (\psi(Y) \wedge \psi(Z) \wedge x \in Y \cap Z) \rightarrow Y \subseteq Z) \wedge y \in Y, \\ \psi(X) & = & \forall x,y, (x \in X \wedge \bigvee_{c \in \Sigma} E_c(x,y)) \rightarrow y \in X. \end{array}$$

Intuitivement la formule $\varphi(x,y)$ exprime que y appartient au plus petit ensemble de sommets contenant x et clos par les arcs du graphe. Pour tout graphe G sur Σ et pour tous sommets u et $v \in V_G$, $G \models \varphi[u,v]$ si et seulement si il existe un chemin de u à v dans G.

La théorie monadique d'un graphe G est l'ensemble des énoncés de MSO sur la signature de G et satisfaits par G. Un graphe G a une théorie monadique décidable si sa théorie monadique est récursive.

Deux graphes notables ayant une théorie monadique décidable sont la demi-droite Δ_1 [Büc62] et l'arbre binaire complet Δ_2 [Rab69] qui sont représentés à la figure 1.3. Dans ces deux cas, le résultat de décidabilité est obtenu en établissant une correspondance entre un certain modèle d'automates s'exécutant sur des coloriages de ces graphes et les formules de la logique monadique. Les automates capturant MSO sur les arbres déterministes sont présentés dans le paragraphe suivant.

Un exemple classique de graphe dont la théorie monadique est indécidable est la grille infinie, notée Grid, présentée à la figure 1.3. La preuve consiste à construire pour toute machine M déterministe à deux compteurs et tests à zéro un énoncé monadique φ_M telle Grid $\models \varphi_M$ si et seulement si M s'arrête. Comme le problème de l'arrêt de ces machines est indécidable [Min67], la théorie monadique de Grid est indécidable.

1.4.4 Automates d'arbres avec conditions de parité

Sur les arbres déterministes colorés, la logique monadique peut être caractérisée par des automate d'arbres avec conditions de parité.

Définition 1.4.3. Un *automate d'arbres avec conditions de parité* (ou simplement automate d'arbres à parité) sur les (Σ,C)-arbres déterministes est donné par un uplet (Q,I,Δ,Ω) où Q est l'ensemble fini des états, $I \subseteq Q$ est l'ensemble des états initiaux, $\Delta \subseteq Q \times 2^C \times (\Sigma \dashrightarrow Q)$ est l'ensemble des transitions et où Ω est une fonction de Q dans $\mathbb{N}$.

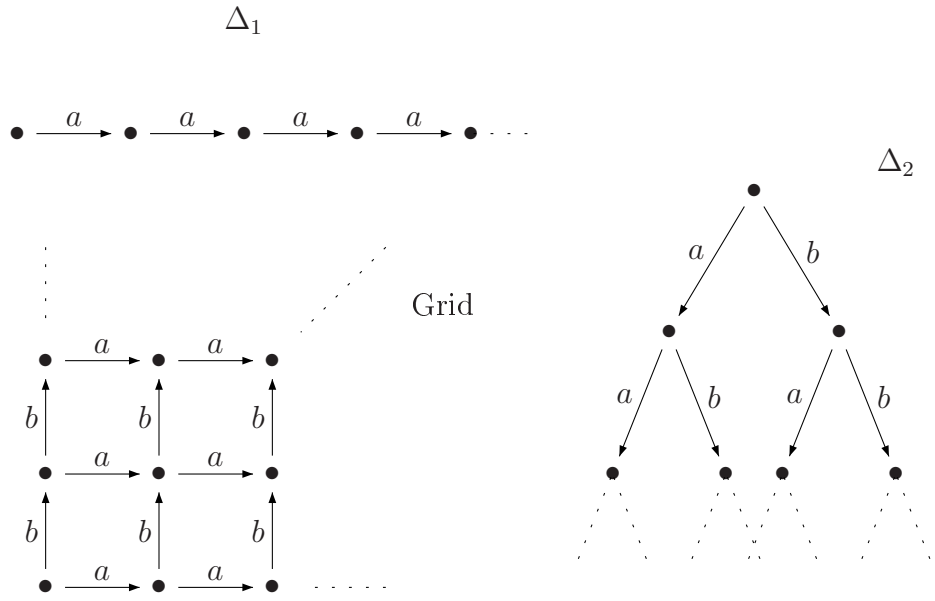


FIG. 1.3 – *En haut à gauche, la demi-droite* Δ_1. *En bas à gauche, la grille infinie Grid. À droite, l'arbre binaire complet* Δ_2.

Une exécution ρ d'un automate d'arbres à parité $A = (Q,I,\Delta,\Omega)$ sur un (Σ,C)-arbre déterministe t est une fonction de $\mathrm{Dom}(t)$ dans Q telle que $\rho(\varepsilon) \in I$ et telle que pour tout $u \in \mathrm{Dom}(t)$ la transition $\delta_u = (\rho(u),t(u),f_u)$, où f_u est la fonction partielle de Σ dans Q définie par $f_u(a) = \rho(ua)$ pour $a \in \Sigma$, appartiennent à Δ.

Nous noterons Φ_ρ la fonction de $\mathrm{Dom}(\rho)$ dans Δ qui à chaque nœud $u \in \mathrm{Dom}(t) = \mathrm{Dom}(\rho)$ associe la transition δ_u. Nous dirons que l'exécution part de l'état q (resp. part de la transition $\delta_0 \in \Delta$) si la condition $\rho(\varepsilon) \in I$ est remplacée par $\rho(\varepsilon) = q$ (resp. par $\Phi_\rho(\varepsilon) = \delta_0$).

De plus, une exécution ρ de A sur t est acceptante si pour toute branche infinie $\pi = (u_i)_{i\geq 0}$ de t, le plus petit entier apparaissant infiniment souvent dans $(\Omega(\rho(u_i)))_{i\geq 0}$ est pair. S'il existe une exécution acceptante de A sur t, nous dirons simplement que A accepte t.

Dans [Rab69], Rabin établit la correspondance entre la logique monadique et les automates d'arbres avec conditions d'acceptation de Rabin pour les arbres déterministes et complets colorés. La condition de parité a été introduite indépendamment dans [EJ91, Mos91]. L'extension aux arbres déterministes mais non complets est classique; pour une preuve complète de ce résultat, on pourra voir

par exemple [Tho97].

Théorème 1.4.4 ([Rab69]). *Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Omega$ et pour tout énoncé φ de MSO, il existe un automate d'arbres à parité A_φ sur les (Σ,C)-arbres déterministes tel que pour tout arbre déterministe t sur (Σ,C), $t \models \varphi$ si et seulement si A_φ accepte t.*

Remarquons que la construction de l'automate à partir de la formule est effective.

Ce théorème peut être formulé pour prendre en compte les formules de MSO avec des variables libres. Pour toute formule $\varphi(X_1,\ldots,X_n)$, il existe un automate d'arbre à parité $A = (Q,I,\Delta,\Omega)$ avec une famille $(Q_i)_{i\in[1,n]}$ de sous-ensembles de Q tel que pour tout arbre déterministe t:

- s'il existe une exécution acceptante ρ de A sur t alors $t \models \varphi[U_1,\ldots,U_n]$ où pour tout $i \in [1,n]$, U_i est l'ensemble des nœuds u telle que $\rho(u) \in Q_i$,
- réciproquement, si pour des sous-ensembles $U_1,\ldots,U_n$ de $\mathrm{Dom}(t)$, $t \models \varphi[U_1,\ldots,U_n]$ alors il existe une exécution acceptante ρ de A sur t telle que pour tout $i \in [1,n]$, U_i soit l'ensemble des nœuds u de t tels que $\rho(u) \in Q_i$.

Dans le chapitre 3, nous nous intéressons à des formules monadiques ayant une ou deux variables libres du premier ordre. Ces formules interviennent dans la définition des interprétations monadiques présentées dans ce chapitre. Nous présentons maintenant une forme d'automates adaptée à ces formules. Pour toute formule monadique $\varphi(x_0,x_1)$, il existe un automate d'arbres à parité A dont l'ensemble des états est de la forme $Q\times 2^{\{0,1\}}\times 2^{\{0,1\}}$ tel que toute exécution acceptante ρ de A sur t satisfasse:

- pour tout $i \in \{0,1\}$, il existe un unique nœud u_i tel que $\rho(u_i) = (q,M,R)$ avec $i \in M$,
- pour tout $i \in \{0,1\}$ et pour tout nœud v avec $\rho(v) = (q,M,R)$, $i \in R$ si et seulement si $v \sqsubset u_i$,
- et $t \models \varphi[u_0,u_1]$.

1.4.5 Jeux de parité

Dans ce paragraphe, nous introduisons les jeux de parité (cf. sous-paragraphe 1.4.5.1) et leur liens avec les automates d'arbres à parité (cf. sous-paragraphe 1.4.5.2). Nous concluons en rappelant quelques résultats liant la solution des jeux de parité et la logique monadique (cf. sous-paragraphe 1.4.5.3).

1.4.5.1 Définition et propriétés

Un *jeu de parité* est un graphe G sur $(\Sigma,\{0,1,p_0,\ldots,p_N\})$ pour $N \geq 0$ tel que pour tout $u \in V_G$, $\Theta_G(u) = \{i_u,p_u\}$ avec $i_u \in \{0,1\}$ et $p_u \in \{p_0,\ldots,p_N\}$. Le

joueur 0 (resp. joueur 1) joue sur les sommets colorés par 0 (resp. colorés par 1). Nous noterons V_i l'ensemble des sommets colorés par i pour $i \in \{0,1\}$. De plus, nous supposerons toujours que G est un sous-ensemble de $V_0 \times \Lambda \times V_1 \cup V_1 \times \Lambda \times V_0$. Remarquons que les étiquettes ne jouent aucun rôle dans le jeu mais elles nous seront utiles pour exprimer des propriétés des ces jeux en logique monadique dans le sous-paragraphe 1.4.5.3.

Une *partie* est un chemin fini ou infini dans le graphe G. Une partie finie π à partir d'un sommet $u \in V_G$ jusqu'à un sommet $v \in V_G$ est gagnée par le joueur i si v appartient à V_{1-i} et si le degré sortant de v est égal à 0. Une partie infinie $(u_j a_j)_{j \in \mathbf{N}}$ est gagné par le joueur 0 (resp. le joueur 1) si le plus petit nombre apparaissant infiniment souvent dans $(p_{u_j})_{j \geq \mathbf{N}}$ est pair (resp. impair).

Une *stratégie* Φ *pour le joueur* i est une fonction partielle de l'ensemble des chemins finis sur G terminant dans V_i et à valeur dans V_{1-i} telle que pour tout chemin fini π terminant en $v \in V_i$ appartenant à $\mathrm{Dom}(\Phi)$, $(v,a,\Phi(\pi)) \in G$ pour un certain $a \in \Lambda_G$. Une partie finie $u_0 a_1 u_1 \dots a_n u_n$ suit la stratégie Φ si pour tout $j \in [1,n]$, $v_j \in V_{i-1}$ implique $v_j = \Phi(\pi_j)$ où $\pi_j = u_0 a_1 \dots u_{j-1}$. Une partie infinie suit la stratégie Φ si tous ses préfixes finis suivent Φ.

Une stratégie Φ pour le joueur i est *positionnelle* si elle ne dépend que du dernier sommet du chemin (*i.e.* pour tous chemins finis π et π' terminant en $v \in V_i$, $\Phi(\pi) = \Phi(\pi')$). Une stratégie positionnelle pour le joueur i est entièrement décrite par une fonction partielle de V_i dans V_{i-1}.

Une stratégie Φ pour le joueur i est gagnante à partir de $u \in V_G$ si toute partie commençant en u et suivant Φ est gagnée par le joueur i. Nous dirons que le joueur i gagne G depuis u s'il existe une stratégie gagnante pour i à partir de u. La région gagnante du joueur i, notée W_i, est l'ensemble des sommets de G à partir desquels le joueur i gagne le jeu. D'après [Mar75], nous savons que les jeux de parité sont déterminés (*i.e.* $V_G = W_0 \cup W_1$).

La propriété fondamentale des jeux de parité est qu'ils peuvent être gagnés en utilisant uniquement des stratégies positionnelles. Ce résultat a été obtenu par Mostowski dans [Mos91] et par Emmerson et Julta dans [EJ91].

Théorème 1.4.5. *Pour tout jeu de parité G et pour tout $u \in V_G$, le joueur 0 ou le joueur 1 possède une stratégie positionnelle gagnante à partir de u.*

1.4.5.2 Lien avec les automates d'arbres à parité

Ce sous-paragraphe est adapté de [Wal02]. À partir d'un automate d'arbres à parité A et d'un arbre déterministe t, nous construisons un jeu de parité G_A^t tel que le joueur 0 gagne le jeu à partir du sommet u_0 si et seulement si A accepte t. Considérons un automate d'arbres à parité $A = (Q,I,\Delta,\Omega)$ sur les (Σ,C)-arbres déterministes et un arbre déterministe t sur (Σ,C). Soit $N \geq 0$ tel que $\Omega(Q) \subseteq [0,N]$.

Considérons le jeu de parité G_A^t étiqueté par l'ensemble $\Sigma \cup \Delta$ et coloré par $\{0,1,p_0,\ldots,p_N\}$.

$$\begin{array}{rcl} G_A^t & = & \{((q,u),\delta,(\delta,u)) \in V_0 \times \Delta \times V_1 \mid \delta = (q,f)\} \\ & \cup & \{((\delta,u),a,(q,ua)) \in V_1 \times \Sigma \times V_0 \mid \delta = (p,f), a \in \mathrm{Dom}(f) \text{ et } q = f(a)\} \\ & \cup & \{(p_i,v),(0,v) \mid v = (q,u) \in V_0 \cap V_{G_A^t} \text{ et } i = \Omega(q)\} \\ & \cup & \{(p_i,v),(1,v) \mid v = ((q,f),u) \in V_1 \cap V_{G_A^t} \text{ et } i = \Omega(q)\} \end{array}$$

où $V_0 = Q \times \mathrm{Dom}(t)$ et $V_1 = \{(\delta,u) \mid \delta = (q,f), \forall a\Sigma, ua \in \mathrm{Dom}(t) \Leftrightarrow a \in \mathrm{Dom}(f)\} \subseteq \mathrm{Dom}(t) \times \Delta$.

Par construction, le joueur 0 gagne G_A^t depuis $(q_0,\varepsilon) \in V_G$ si et seulement si A accepte t. Plus précisément, toute stratégie positionnelle Φ pour le joueur 0 depuis (q,ε) (resp. depuis (δ,ε)) induit une exécution acceptante de A sur t commençant par q (resp. par δ) telle $\Phi(\rho(u),u) = (\Phi_\rho(u),u)$ et vice-versa.

Une conséquence du théorème 1.4.5 est que nous pouvons restreindre notre attention aux exécutions régulières des automates d'arbres à parité. Une exécution ρ d'un automate A sur un arbre t est *régulière* si pour tous nœuds u et v différents de la racine ε, $t_{/u} \approx t_{/v}$ et $\rho(u) = \rho(v)$ implique $\Phi_\rho(u) = \Phi_\rho(v)$. La proposition suivante est tirée de [Wal02].

Lemme 1.4.6 ([Wal02]). *Si un automate d'arbres à parité accepte un arbre t alors il existe une exécution acceptante régulière de $A = (Q,I,\Delta,\Omega)$ sur t. Plus précisément, pour toute transition $\delta \in \Delta$, s'il existe une exécution acceptante de A commençant par δ alors il existe une exécution acceptante régulière commençant par δ.*

Démonstration. Considérons la relation d'équivalence R sur $\mathrm{Dom}(t)$ définie pour tout $u,v \in \mathrm{Dom}(t)$ par $(u,v) \in R$ si et seulement si $t_{/u} \approx t_{/v}$. Pour tout $v = (q,u) \in V_0$, nous noterons $[v]_R = (q,[u]_R)$ et pour tout $v = (\delta,u) \in V_1$, nous noterons $[v]_R = (\delta,[u]_R)$.

Le quotient de G_A^t par R est le jeu $\overline{G_A^t}$.

$$\begin{array}{rcl} \overline{G_A^t} & = & \{([u]_R,a,[v]_R) \mid (u,a,v) \in G_A^t\} \\ & \cup & \{(c,[u]_R) \mid (c,u) \in G\} \end{array}$$

Pour tout $v \in V_{G_A^t}$, le joueur 0 gagne G_A^t depuis v si et seulement si il gagne $\overline{G_A^t}$ depuis $[v]_R$. En effet, le dépliage de G_A^t depuis v est isomorphe au dépliage de $\overline{G_A^t}$ depuis $[v]_R$.

Supposons qu'il existe une exécution acceptante de A sur t depuis δ_0. Le joueur 0 gagne G_A^t depuis (δ_0,ε) et donc il gagne $\overline{G_A^t}$ depuis $(\delta_0,[\varepsilon]_R)$. Soit Φ une stratégie gagnante positionnelle sur $\overline{G_A^t}$ pour le joueur 0 à partir de $(\delta_0,[\varepsilon]_R)$ (cf. théorème 1.4.5). Cette stratégie induit une exécution acceptante régulière ρ de A sur t qui satisfait $\Phi_\rho(\varepsilon) = \delta_0$ et pour tout $u \neq \varepsilon$, $\Phi_\rho(u) = \delta$ avec $\Phi((\rho(u),[u]_R)) = (\delta,[u]_R)$. Par définition de R, ρ est une exécution régulière. □

1.4.5.3 Lien avec MSO

Dans [Wal02], l'auteur établit que la région gagnante pour le joueur i sur un jeu G est définissable en logique monadique.

Proposition 1.4.7. *Pour tout jeu G, la région gagnante du joueur 0 (resp. joueur 1) est définissable en logique monadique.*

Si le graphe définissant le jeu est déterministe, nous pouvons affiner ce résultat en donnant une formule qui exprime l'existence d'une stratégie positionnelle gagnante. Comme G est déterministe, une stratégie positionnelle pour le joueur i est décrite par une fonction de V_i dans $\Lambda_G = \{ a_1, \ldots, a_n \}$. Une famille $(U_j)_{j \in [1,n]}$ de sous-ensembles deux à deux disjoints de V_G définit une stratégie positionnelle pour le joueur i si l'union des U_j est égale à V_i et si pour tout $u \in U_j$, il existe $v_u \in V_G$ tel que $(u,a_j,v_u) \in G$. La stratégie positionnelle associée Φ est définie par $\Phi(u) = v_u$. Comme G est déterministe, la logique monadique est équivalente à la logique monadique gardée: un enrichissement de la logique monadique où l'on autorise la quantification sur les ensembles d'arcs [Cou03]. Une conséquence de ce résultat est que nous pouvons exprimer en logique monadique l'existence d'une stratégie positionnelle gagnante à partir d'un sommet donné. Un construction explicite de cette formule est donnée dans [Cac03a, p. 17].

Proposition 1.4.8. *Pour tout jeu déterministe G étiqueté par $\Lambda_G = \{a_1, \ldots, a_n\}$ et coloré $\Theta_G = \{0,1,p_0, \ldots, p_N\}$ et pour tout $i \in \{ 0,1 \}$, il existe une formule monadique $\Psi_i(x,X_1, \ldots, X_n)$ telle que pour tout $u \in V_G$ et pour tout $U_1, \ldots, U_n \subseteq V_G$, $G \models \Psi_i[u,U_1, \ldots, U_n]$ si et seulement si les ensembles $U_1, \ldots, U_n$ définissent une stratégie positionnelle Φ pour le joueur i gagnante à partir de u.*

1.4.6 Propriété de sélection pour MSO

Dans ce sous-paragraphe, nous présentons la propriété de sélection pour la logique monadique. Cette notion est «orthogonale» à la décidabilité de la théorie monadique. Cette propriété permet dans le cas où un graphe G satisfait à une formule existentielle $\exists X, \varphi(X)$ de définir en logique monadique un ensemble de sommets U tel que $G \models \varphi[U]$.

Un graphe G satisfait la propriété de sélection si pour toute formule $\varphi(X)$, nous pouvons effectivement construire une formule $\psi(X)$ telle que:

$$\begin{cases} G & \models & \forall X, \psi(X) \rightarrow \varphi(X) \\ G & \models & (\exists X, \psi(X)) \leftrightarrow (\exists X, \varphi(X)) \\ G & \models & \exists^{\leq 1} X, \psi(X) \end{cases}$$

La formule $\psi(X)$ est appelée un *sélecteur* de la formule $\varphi(X)$ sur G. La notion de sélecteur est immédiatement étendue aux formules possédant plusieurs variables

libres. En fait, si nous pouvons construire des sélecteurs pour toutes les formules ayant une variable libre, nous pouvons aussi construire des sélecteurs pour les formules ayant un nombre arbitraire de variables libres.

Proposition 1.4.9. *Si un graphe G possède la propriété de sélection alors pour tout $n \geq 1$ et pour tout formule $\varphi(X_1,\ldots,X_n)$, nous pouvons construire un sélecteur $\psi(X_1,\ldots,X_n)$ de $\varphi(X_1,\ldots,X_n)$ sur G.*

Démonstration. Soit G un graphe ayant la propriété de sélection. Nous établissons la propriété par récurrence sur $n \geq 1$. Le cas de base $n = 1$ est immédiat. Supposons que la propriété est établie par $n \geq 1$ et montrons qu'elle est vraie pour $n+1$. Soit $\varphi(X_1,\ldots,X_n,X_{n+1})$ une formule monadique. Considérons la formule $\varphi_0(X_1,\ldots,X_n) = \exists X_{n+1}, \varphi(X_1,\ldots,X_n,X_{n+1})$. Par hypothèse de récurrence, nous pouvons construire un sélecteur $\psi_0(X_1,\ldots,X_n)$ de φ_0 sur G. Considérons maintenant la formule $\varphi_1(X) = \exists X_1,\ldots,X_n, \psi_0(X_1,\ldots,X_n) \wedge \varphi(X_1,\ldots,X_n,X)$. Comme G a la propriété de sélection, nous pouvons construire un sélecteur $\psi_1(X)$ de φ_1 sur G. Nous vérifions aisément que la formule $\psi(X_1,\ldots,X_{n+1})$ définie par:

$$\psi(X_1,\ldots,X_{n+1}) = \psi_0(X_1,\ldots,X_n) \wedge \psi_1(X_{n+1}).$$

est un sélecteur de $\varphi(X_1,\ldots,X_{n+1})$ sur G. □

Dans [LS98], Lifsches et Shelah montrent que tous les coloriages de la demi-droite possèdent la propriété de sélection et ce indépendamment de la décidabilité de la théorie monadique.

Dans [Rab69], l'auteur établit, sur l'arbre binaire complet Δ_2 étiqueté par $\{a,b\}$ (cf. figure 1.3), la conséquence suivante du théorème 1.4.4.

Théorème 1.4.10. *Pour toute formule monadique $\varphi(X)$, si l'arbre binaire complet Δ_2 satisfait $\exists X, \varphi(X)$ alors il existe un ensemble rationnel $R_\varphi \in Rat(\{a,b\}^*)$ tel que $\Delta_2 \models \varphi[R_\varphi]$. De plus, un automate fini acceptant R_φ peut être effectivement construit à partir de φ.*

Comme l'accessibilité depuis la racine par un chemin appartenant à un ensemble rationnel est définissable en logique monadique, le théorème précédent implique que Δ_2 possède la propriété de sélection.

Chapitre 2

Automates infinis

Le terme d'*automate infini* a été introduit dans [Tho01]. Il désigne un graphe infini de présentation finie. Ceci signifie que ce graphe infini est décrit par une quantité finie d'informations. L'emploi du terme d'automate au lieu de celui de graphe souligne le fait que nous nous intéressons aux langages acceptés par ces graphes. À tout graphe infini G étiqueté par Σ, nous pouvons associer un langage de mots sur Σ en fixant un ensemble $I \subseteq V_G$ de sommets initiaux et un ensemble $F \subseteq V_G$ de sommets finaux. La trace de G entre I et F, notée $\mathcal{L}(G,I,F)$, est l'ensemble des étiquettes des chemins de G partant d'un sommet $i \in I$ et arrivant à un sommet $f \in F$.

Un point important est que, dans cette approche, nous considérons les graphes à isomorphisme près. Autant que possible, nous cherchons à donner des propriétés qui soient indépendantes de l'ensemble des sommets choisis pour définir le graphe. Le langage accepté par un automate infini permet donc de donner une mesure «grossière» de la complexité de sa structure. Une deuxième mesure est donnée par l'expressivité des logiques décidables sur cet automate.

Dans ce chapitre, nous présentons des familles d'automates infinis acceptant les différents niveaux de la hiérarchie de Chomsky. Nous suivons l'approche présentée dans [CK02] où les auteurs présentent une hiérarchie d'accepteurs infinis similaire à la hiérarchie de Chomsky. Cette hiérarchie d'accepteurs infinis est présentée à la figure 2.1.

Dans le paragraphe 2.1, nous introduisons les différents types de présentation finie qui apparaissent dans la littérature. En particulier, nous définissons les différents graphes associés aux systèmes de transitions étiquetées: graphes enracinés, graphes des configurations et graphes des transitions.

Dans le paragraphe 2.2, nous présentons les différentes familles de graphes définies autour des automates à pile: les graphes enracinés des automates à pile [MS85], les graphes des configurations des automates à pile [Cau92], les graphes HR-equationnels [Cou89] et les graphes préfixe-reconnaissables [Cau96].Toutes

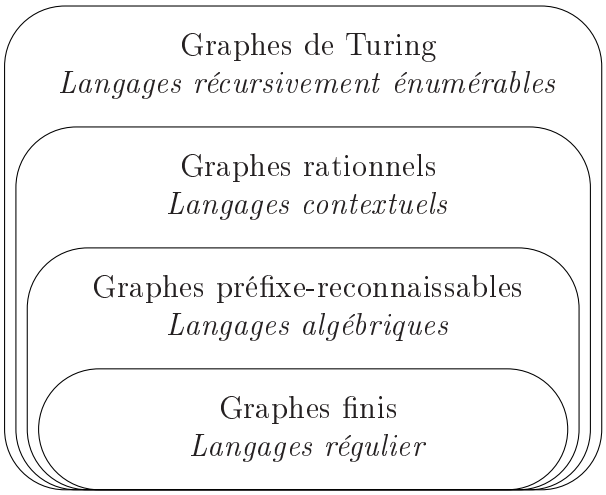


FIG. 2.1 – *Une hiérarchie à la Chomsky de graphes infinis.*

ces familles d'automates infinis ont pour traces les langages algébriques.

Dans le paragraphe 2.3, nous présenterons différentes familles d'automates infinis ayant pour traces les langages contextuels. Ce paragraphe sera l'occasion de donner un résumé des travaux effectués durant ma thèse en collaboration avec Antoine Meyer autour de ces familles. Ces travaux se composent de deux parties. Dans la première partie, nous nous sommes intéressés aux graphes rationnels [Mor00, MS01] et à leurs sous-familles. Ces travaux en commun ont été publiés dans [CM06a].
Dans la deuxième partie, nous avons étudié les graphes de transitions des machines de Turing linéairement bornées (cf. sous-paragraphe 1.2.3) et en particulier leurs liens avec les graphes rationnels. Une version préliminaire de cette étude a été publiée dans [CM05] et une version complète a été publiée dans [CM06b]. Tous ces travaux sont présentés dans la thèse d'Antoine Meyer [Mey05, ch. 7].

2.1 Présentations finies

En suivant [Cau96], nous distinguons deux types de présentations finies. Les premières, dites *internes*, fixent un nommage des sommets du graphe et décrivent

explicitement et avec une quantité finie d'informations les arcs du graphe. Les présentations internes sont toujours associées à des machines ou à des automates (cf. sous-paragraphe 2.1.1). Les secondes, dites *externes*, définissent la structure du graphe à isomorphisme près. Nous présentons brièvement deux approches externes (cf. paragraphe 2.1.2). L'une consiste à exprimer le graphe comme la plus petite solution d'un système d'équations et l'autre à définir le graphe par une suite de transformations appliquées à un graphe de présentation finie.

2.1.1 Graphes définis par des machines

Nous présentons les différents types de graphes associés à des machines ayant un nombre finis d'états.

2.1.1.1 Graphes de calcul

La manière la plus simple de donner une présentation finie d'un graphe est de fixer un ensemble de sommets V. Dans le cadre de ce document, V sera toujours un ensemble de mots sur un alphabet fini. Il est cependant possible de considérer des ensembles de sommets plus riches comme par exemple des termes finis. Une fois l'ensemble des sommets fixés, il faut pour définir un graphe G étiqueté par Σ donner une famille de relations finies $(R_a)_{a\in\Sigma}$ sur V. Un graphe G étiqueté par Σ est alors décrit par une famille de machines $\mathcal{T} = (T_a)_{a\in\Sigma}$ acceptant des relations sur V. Le graphe défini par cette famille est:

$$G_{\mathcal{T}} = \{(v,a,v') \mid v,v' \in V \text{ et } (v,v') \text{ accepté par } T_a\}.$$

Les graphes rationnels (cf. paragraphe 2.3.1) et les graphes préfixe-reconnaissables (cf. paragraphe 2.2) sont des exemples de graphes de calcul.

2.1.1.2 Graphes associés aux LTS

Dans le paragraphe 1.2, nous avons donné pour chaque niveau de la hiérarchie de Chomsky des systèmes de transitions étiquetées associés aux accepteurs finis de ce niveau. Il est donc naturel de considérer les différents types de graphes associés à ces systèmes.

Un système de transitions étiquetées par Σ peut être vu comme un graphe étiqueté par $\Sigma \cup \{\tau\}$ dont l'ensemble des sommets est l'ensemble des configurations du système et qui est défini par:

$$\{(c,x,c') \mid c,c' \in \mathcal{C}, x \in \Sigma \cup \{\tau\} \text{ et } c \xrightarrow{x} c'\}.$$

Ce graphe canonique sera dans la suite appelé graphe du LTS.

Pour faire apparaître le comportement du système, il est, en général, nécessaire d'opérer une restriction sur l'ensemble des configurations et de masquer les transitions internes (*i.e.* étiquetées par τ) du LTS. Si nous laissons apparentes les transitions internes du système, nous obtenons les notions de graphe enraciné ou de graphe des configurations. En «masquant» ces transitions, nous obtenons la notion de graphe des transitions.

Graphes enracinés Le graphe le plus naturel associé à un LTS est obtenu en fixant une configuration initiale c_0 et en ne conservant que les configurations accessibles depuis c_0. Plus précisément, pour tout système de transitions étiquetées $\mathcal{S} = (\xrightarrow{x})_{x\in\Sigma\cup\{\tau\}}$ avec un ensemble de configurations $\mathcal{C}$ et pour toute configuration $c_0 \in \mathcal{C}$, le graphe de $\mathcal{S}$ enraciné en c_0, noté $R_{\mathcal{S}}^{c_0}$, est défini par:

$$R_{\mathcal{S}}^{c_0} = \{(c,x,c') \mid x \in \Sigma \cup \{\tau\}, c_0 \longrightarrow^* c \text{ et } c \xrightarrow{x} c'\}.$$

Nous écrirons simplement $R_{\mathcal{S}}$ si c_0 se déduit du contexte.

Graphes des configurations Les graphes enracinés présentés ci-dessus fournissent une restriction naturelle de l'ensemble des configurations qui ne dépend pas de la nature du LTS. Cependant cette notion impose de travailler avec des graphes dont tous les sommets sont accessibles à partir d'un sommet donné.

D'un point de vue structurel, une notion plus pertinente est obtenue en restreignant le graphe du LTS à un ensemble «régulier» de configurations. La notion de régularité doit être précisée pour chaque type de LTS. Les graphes ainsi définis sont appelés les graphes des configurations.

Graphes des transitions Le graphe des transitions est obtenu en masquant les transitions étiquetées par τ dans le graphe des configurations du LTS. La définition que nous présentons est tirée de [Sti00]. Pour assurer que la suppression des transitions étiquetées par τ préserve le langage accepté par le système, il nous faut nous restreindre à des LTS dits *normalisés*.

Nous dirons qu'une configuration $c \in \mathcal{C}$ est *observable* si elle n'est la source d'aucune transition étiquetée par τ. Nous noterons $\mathcal{C}_{\text{obs}} \subseteq \mathcal{C}$ l'ensemble des configurations observables. Une configuration $c \in \mathcal{C}$ est dite *interne* si elle est la source d'au moins une transition étiquetée par τ et si elle n'est la source d'aucune transition étiquetée dans Σ. Nous noterons $\mathcal{C}_{\text{int}} \subseteq \mathcal{C}$ l'ensemble des configurations internes. Un système de transitions étiquetées est dit normalisé si toutes ses configurations sont soit observables, soit internes *i.e.* $\mathcal{C} = \mathcal{C}_{\text{obs}} \cup \mathcal{C}_{\text{int}}$.

Exemple 2.1.1. La figure 2.2 présente deux LTS $\mathcal{S}_1$ et $\mathcal{S}_2$ étiquetés par $\Sigma = \{a,b\}$. Le système $\mathcal{S}_1$ n'est pas normalisé car la configuration c_1 n'est ni observable

(car $c_1 \xrightarrow{\tau} c_2$) ni interne (car $c_1 \xrightarrow{b} c_4$). Le système $\mathcal{S}_2$ est normalisé. Les configurations observables sont c_1,c_3 et c_4 et l'unique configuration interne est c_2.

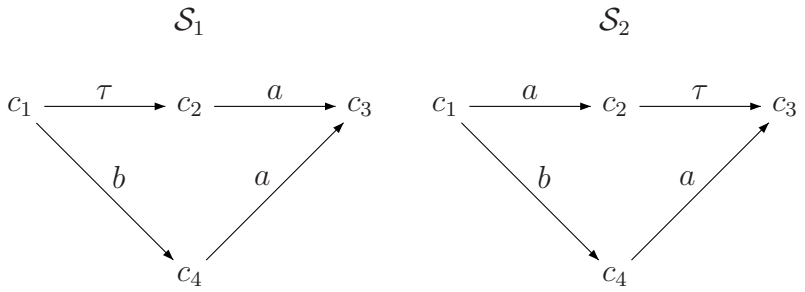


FIG. 2.2 – *Deux systèmes de transitions étiquetées $\mathcal{S}_1$ et $\mathcal{S}_2$.*

Le graphe des transitions, noté $G_{\mathcal{S}}$, d'un système de transitions étiquetées normalisé $\mathcal{S}$ est défini en se restreignant aux configurations observables de son graphe des configurations et en mettant un arc étiqueté par $a \in \Sigma$ entre c et $c' \in \mathcal{C}_{\text{obs}}$ s'il existe un chemin dans $C_{\mathcal{S}}$ partant de c et arrivant en c' étiqueté par $a\tau^*$.

$$G_{\mathcal{S}} = \{(c,a,c') \mid c,c' \in \mathcal{C}_{\text{obs}}, a \in \Sigma \text{ et } c \xrightarrow[C_{\mathcal{S}}]{a\tau^*} c'\}.$$

En terme de transformation de graphes, la transformation passant du graphe des configurations au graphe des transitions sera appelée la τ-fermeture.

Exemple 2.1.2. La figure 2.3 présente le graphe d'un système de transitions étiquetées normalisé $\mathcal{S}$ enraciné en c_1 et son graphe des transitions $G_{\mathcal{S}}$. Remarquons qu'un système de transitions étiquetées non déterministe peut induire un graphe des transitions déterministe.

2.1.2 Présentations externes

Les présentations externes décrivent la structure du graphe à isomorphisme près et ne fournissent pas un nommage explicite des sommets.

2.1.2.1 Systèmes d'équations

Dans cette approche, les graphes infinis sont définis comme la solution d'un système fini d'équations. Pour définir cette notion, il faut un jeu d'opérateurs sur les graphes. Les deux jeux usuels sont les opérateurs HR et VR. Pour une présentation détaillée, nous renvoyons le lecteur à [Cou97].

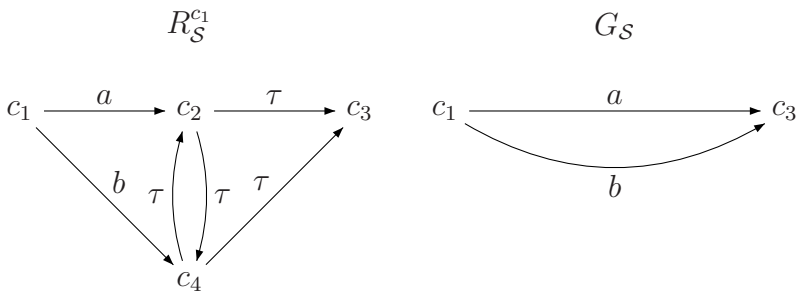


FIG. 2.3 – *Graphe enraciné d'un LTS (à gauche) et le graphe des transitions correspondant (à droite).*

Les opérateurs HR consistent en l'ensemble des graphes finis colorés, l'union disjointe, le recoloriage (cf. paragraphe 3.2) et la fusion des sommets possèdant la même couleur. Une vision alternative naturelle des systèmes finis d'équations sur les opérateurs HR est donnée par les grammaires déterministes de graphes que nous présenterons dans le sous-paragraphe 2.2.2.

Les opérateurs VR sont le sommets isolés colorés, l'union disjointe, l'ajout d'un arc étiqueté par $a \in \Sigma$ de tout sommet coloré par $c_1 \in C$ à tout sommet colorié par $c_2 \in C$ et le recoloriage.

Les graphes colorés (sur un univers fixé) forment pour l'inclusion un *cpo* (complete partial order) pour lequel les opérateurs VR sont continus. Il suit donc que tout système fini d'équation sur ces opérateurs admet une plus petite solution. Un graphe est dit VR-équationnel s'il est isomorphe à la plus petite solution d'un système fini d'équations sur les opérateurs VR.

Les opérateurs VR ont été augmentés par le produit asynchrone et le produit synchronisé pour obtenir les familles de graphes VRA-équationnels et VRS-équationnels dans [Col04].

2.1.2.2 Transformation de graphes

Nous présenterons, dans le chapitre 3, de nombreuses transformations de graphes qui ont toutes la propriété de préserver la décidabilité de la théorie au second ordre monadique. Nous renvoyons le lecteur à ce chapitre pour des définitions précises de ces différentes transformations.

Dans [Cau02], Caucal utilise l'itération de deux de ces transformations: le dépliage et la substitution rationnelle inverse pour définir une hiérarchie de graphes infinis ayant une théorie monadique décidable. Le niveau 0 de cette hiérarchie est la classe des graphes finis. La classe des graphes du niveau $n+1$ est la classe

des graphes isomorphes à un graphe obtenu en appliquant un dépliage[1] suivi d'une substitution rationnelle inverse à un graphe du niveau n. Cette hiérarchie de graphes infinis est présentée en détails dans le chapitre 5.

Une autre approche par transformations de graphes est développée par Colcombet et Löding dans [CL06] où les auteurs considèrent une transformation de graphes qui partant d'un graphe ayant une théorie monadique faible[2] décidable, produit un graphe de structure plus riche ayant une théorie au premier ordre décidable. Cette transformation est une variante de l'interprétation monadique dans laquelle les sommets du graphe d'arrivée sont des ensembles finis de sommets du graphe de départ.

Cette approche est très riche mais ne peut pas être itérée puisque les graphes obtenus ont uniquement une théorie au premier ordre décidable et non plus une théorie monadique faible décidable. Cependant comme les graphes de la hiérarchie définie par Caucal ont une théorie monadique faible décidable, cette transformation permet de définir une hiérarchie de graphes ayant une théorie au premier ordre décidable.

2.2 Autour des automates à pile

2.2.1 Graphes des automates à pile

La première famille d'automates infinis à été étudiée par Muller et Schupp dans [MS85]. Ils considèrent les graphes isomorphes aux graphes enracinés des automates à pile. Ces graphes sont naturellement apparus dans l'étude des graphes de Cayley des groupes algébriques (en anglais *context-free groups*). Les automates à pile qu'ils considèrent sont dit *temps-réel* c'est-à-dire qu'ils ne contiennent pas de transitions étiquetés par τ.

Définition 2.2.1. Les graphes enracinés des automates à pile sont les graphes isomorphes au graphe enraciné d'un automate à pile temps-réel à partir de l'une de ses configurations.

Remarquons que l'on peut supposer sans perte de généralité que cette configuration est la configuration initiale $q_0[]$.

Exemple 2.2.2. Considérons l'automate à pile $P = (\Gamma,\Sigma,Q,q_0,F,\Delta)$ où $\Gamma = \{A,B\}$, $\Sigma = \{a,b\}$, $Q = \{q_0,q_1\}$ et $F = \{q_1\}$ et dont les transitions sont

1. depuis un sommet MSO-définissable.
2. Dans cette variante de la logique monadique, les quantifications ne portent que sur des ensembles finis de sommets.

données par:

$$(q_0,\varepsilon,a,q_0,A) \qquad (q_0,A,a,q_0,AA) \qquad (q_0,A,b,q_1,\varepsilon) \qquad (q_1,A,b,q_1,\varepsilon).$$

Le langage accepté par P est l'ensemble $\{a^n b^m \mid n \geq 1 \text{ et } m \leq n\}$ et son graphe enraciné à partir de la configuration initiale (q_0,ε) est donné dans la figure 2.4.

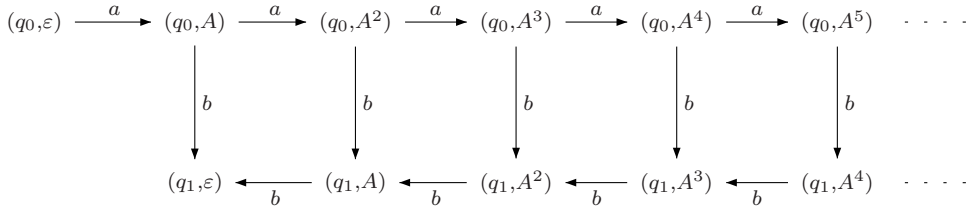


FIG. 2.4 – *Graphe de l'automate à pile P enraciné en (q_0,ε).*

Dans [MS85], les auteurs exhibent une propriété géométrique de cette famille de graphes. Nous qualifions cette propriété de géométrique car elle ne dépend pas du nommage des sommets. Pour ce faire, ils considèrent la décomposition par distance d'un graphe G par rapport à l'un de ses sommets $v_0 \in V_G$. Dans ce cas, la distance entre deux sommets u et $v \in V_G$ est la longueur du plus court chemin non orienté entre u et v. La décomposition par distance est une suite de graphe $(G_{v_0,n})_{n\geq 0}$. Pour $n \geq 0$, $G_{v_0,n}$ désigne le graphe G restreint à l'ensemble des sommets à distance au moins $n+1$ de v_0. La décomposition par distance est dite de type fini si l'ensemble des composantes connexes apparaissant dans l'un des $G_{v_0,n}$ est fini à isomorphisme près. Cette décomposition est illustrée dans la figure 2.5 sur le graphe de l'automate à pile P enraciné en (q_0,ε) et à partir de (q_0,ε) . Il est aisé de vérifier que ce graphe possède une décomposition par distance de type fini à partir de (q_0,ε) car pour tout $n \geq 2$, $G_{(q_0,\varepsilon),n} \approx G_{(q_0,\varepsilon),2}$.

Dans [MS85], les auteurs établissent que les graphes enracinés d'automates à pile admettent une décomposition par distance de type fini à partir de n'importe lequel de leur sommet. Cette propriété leur permet en particulier d'établir que tous ces graphes possèdent une théorie monadique décidable.

Une famille plus large de graphes est obtenue en considérant les graphes des configurations des automates à pile. Ces graphes sont définis dans [Cau92] où ils sont appelés graphes préfixes avec contrôle rationnel. Comme nous l'avons déjà mentionné dans le paragraphe 2.1.1.2 et pour obtenir une notion pertinente de graphe des configurations, il est nécessaire d'introduire une restriction sur

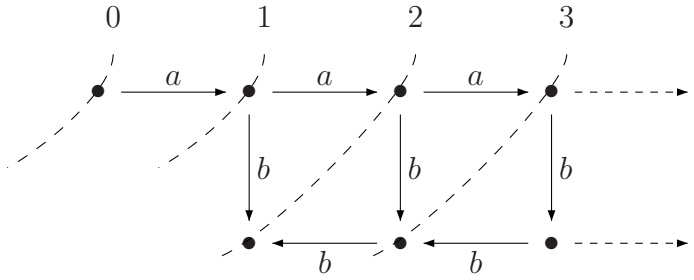


FIG. 2.5 – *Décomposition d'un graphe d'automate à pile par distance depuis sa racine.*

l'ensemble des configurations. Dans le cas des automates à pile, Caucal considère des ensembles rationnels[3] de configurations.

Définition 2.2.3. L'ensemble des graphes des configurations des automates à pile est l'ensemble des graphes isomorphes au graphe d'un automate à pile (temps réel) restreint à un ensemble rationnel de configurations.

Comme l'ensemble des configurations accessibles depuis une configuration donnée forme un ensemble rationnel de configurations (codées comme des mots), les graphes enracinés des automates à pile forment une sous famille des graphes des configurations.

Sur cette famille plus générale, la propriété géométrique de [MS85] permet de caractériser les graphes de configurations des automates à pile.

Théorème 2.2.4 ([MS85],[Cau92])**.** *Les graphes connexes et de degré fini admettant une décomposition finie par distance à partir de chacun de leurs sommets sont les graphes des configurations des automates à pile connexes.*

2.2.2 Graphes HR-equationnels

Les graphes HR-equationnels ont été défini et étudié par Courcelle (cf. [Cou89, Cou90]). Ces graphes peuvent être vus comme les graphes engendrés par les grammaires déterministes de graphes. Une grammaire de graphes généralise aux graphes le principe des grammaires de mots. Pour opérer cette généralisation, il faut considérer non plus des graphes mais des hypergraphes. Intuitivement, un hypergraphe est un graphes dont les arcs peuvent porter sur plus de deux sommets. Ces arcs sont appelés des hyperarcs.

De façon informelle, dans une grammaire de graphes les non-terminaux sont des hyperarcs et les terminaux sont simplement des arcs (*i.e.* des hyperarcs portant sur deux sommets). Une production associe à un hyperarc non-terminal un

3. Une configuration $(q,w) \in \mathcal{C}$ est représentée par le mot $wq \in \Gamma^* Q$.

hypergraphe. Cet hypergraphe va être substitué à l'hyperarc non-terminal dans la dérivation. Pour raccorder l'hypergraphe aux sommets de l'hyperarc dans cette substitution, les sommets de cet hyperarc sont numérotés dans la production à la fois dans le membre gauche et dans le membre droit. Les grammaires considérées sont déterministes c'est-à-dire que chaque hyperarc non-terminal n'apparaît que dans le membre gauche d'une seule production.

Une étape de dérivation consiste à remplacer en parallèle chaque hyperarc non-terminal par l'hypergraphe qu'il lui est associé. Le graphe infini engendré par la grammaire est la limite des étapes successives de dérivation en partant de l'axiome de la grammaire.

Exemple 2.2.5. Dans la figure, nous présentons une grammaire de graphes déterministes qui engendre le graphe présenté dans la figure 2.4. Cette grammaire possède deux non-terminaux S (qui est l'axiome) et A qui sont respectivement d'arité 1 et 2. Ses terminaux sont les arcs étiquetés par a et par b.

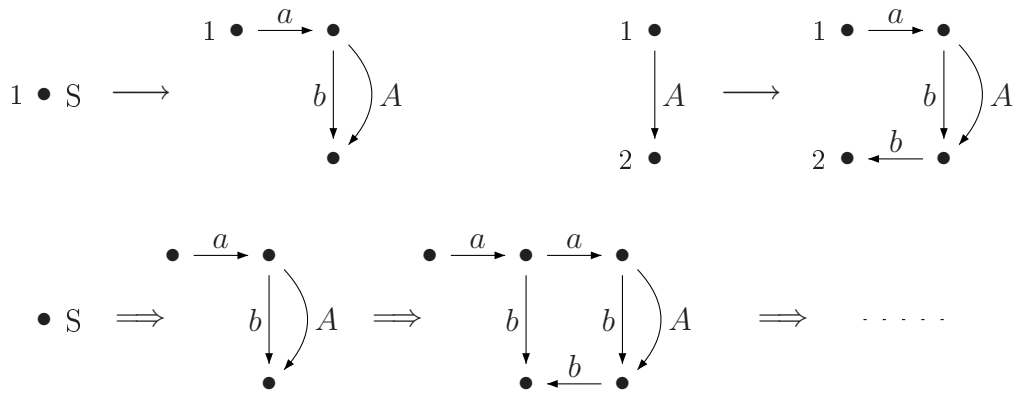


FIG. 2.6 – *Une grammaire de graphes et sa dérivation.*

Dans [Cau95], l'auteur établit que les graphes HR-équationnels contiennent les graphes des configurations des automates à pile. Cette inclusion est stricte. En effet, les graphes des configurations des automates à pile sont de degré borné et comme le montre la figure 2.7, les graphes HR-équationnels peuvent avoir un degré infini. Cependant si l'on ne considère que les graphes de degré fini, les deux familles coïncident [CK01].

2.2.3 Graphes préfixe-reconnaissables

Les graphes préfixe-reconnaissables sont introduits par Caucal dans [Cau96]. Ils sont définis comme les graphes de calcul des relations dites préfixe-reconnaissables sur les mots sur un alphabet fini Γ. Une telle relation est une union finie

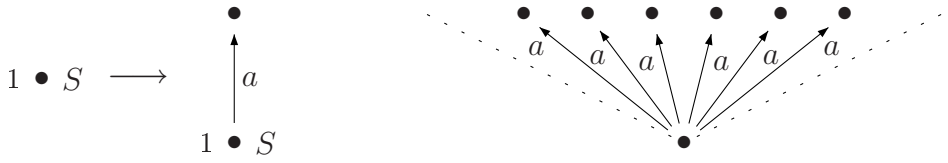


FIG. 2.7 – *Un graphe HR-équationnel de degré infini (à droite) et sa grammaire de graphes déterministe (à gauche).*

---

de relations de la forme $(U \times V) \cdot W$ où U,V et $W \in \mathrm{Rat}(\Gamma^*)$. Les propriétés de ces relations seront rappelées en détail dans le sous-paragraphe 4.5.1.

Un graphe préfixe-reconnaissable étiqueté par Σ est donné par une famille de relations préfixe-reconnaissables $(P_a)_{a\in\Sigma}$ est égal à:

$$\{(u,a,v) \mid a \in \Sigma \text{ et } (u,v) \in P_a\}.$$

Ces graphes admettent de nombreuses présentations internes et externes qui sont résumées dans le théorème ci-dessous:

Théorème 2.2.6. *Les propositions suivantes sont équivalentes à isomorphisme près:*

1. *G est un graphe préfixe-reconnaissable,*
2. *G est un graphe des transitions d'un automate à pile [Sti00],*
3. *G est un graphe obtenu par l'application successive d'un dépliage et d'une substitution rationnelle inverse à un graphe fini [Cau96],*
4. *G est un graphe obtenu par interprétation monadique de l'arbre binaire complet Δ_2 [Bar97, Blu01],*
5. *G est un graphe VR-équationnel [Bar97].*

Les graphes préfixe-reconnaissables ont une théorie monadique décidable. Ceci se déduit de la caractérisation 4 et de la décidabilité de la théorie monadique de l'arbre binaire complet [Rab69]. Leurs traces d'un ensemble fini de sommets à un ensemble fini de sommets sont les langages algébriques.

Les graphes préfixe-reconnaissables contiennent strictement les graphes HR-équationnels et donc les graphes de configurations des automates à pile. L'exem-

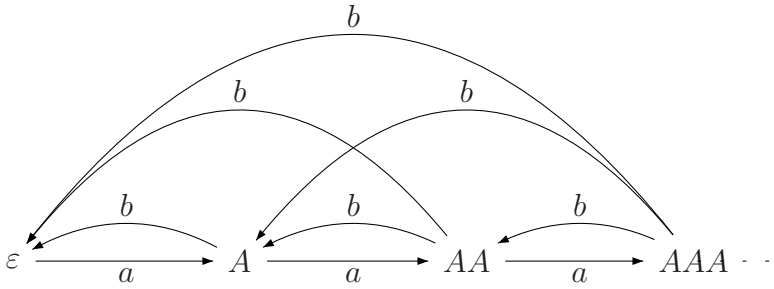


FIG. 2.8 – *Un graphe préfixe-reconnaissable qui n'est pas HR-equationnel.*

ple 2.2.7 présente un graphe préfixe-reconnaissable qui n'est pas (à isomorphisme près) un graphe HR-équationnel.

Cependant ces deux familles peuvent être naturellement caractérisées à l'intérieur des graphes préfixe-reconnaissables. Dans [CK01], les auteurs établissent que les graphes préfixe-reconnaissables de degré borné sont (à isomorphisme près) les graphes des configurations des automates à pile, et les graphes préfixe-reconnaissables ayant un nombre fini de degrés (entrants et sortants) sont (à isomorphisme près) les graphes HR-équationnels.

Exemple 2.2.7. Considérons le graphe préfixe-reconnaissable G étiqueté par $\Sigma = \{a,b\}$ défini par les relations $P_a = (\varepsilon,A) \cdot A^*$ et $P_b = (A^+,\varepsilon) \cdot A^*$. Ce graphe G, qui est présenté dans la figure 2.8, n'est pas un graphe HR-équationnel car pour tout $n \geq 1$ G a un sommet de degré sortant n. Il n'a donc pas une nombre fini de degrés sortants.

2.3 Autour des langages contextuels

Nous présentons dans ce paragraphe deux familles de graphes infinis dont les traces sont les langages contextuels. La première famille est la famille des graphes rationnels définie dans [Mor00]. Cette famille ainsi que certaines de ses sous-familles comme les graphes automatiques [KN94, BG00] sont présentées dans le paragraphe 2.3.1.2. Le paragraphe 2.3.2 présente les graphes linéairement bornés introduits dans [CM05] et étudie leurs relations avec les graphes rationnels.

2.3.1 Graphes rationnels et leurs sous-familles

Les graphes rationnels sont les graphes de calcul des transducteurs de mots. Des sous-familles intéressantes de graphes rationnels sont obtenus en imposant des restrictions sur les transducteurs les définissant. Le paragraphe 2.3.1.1 pré-

sente les transducteurs de mots et certaines sous-familles pertinentes. Le paragraphe 2.3.1.2 présente les graphes rationnels et leurs sous-familles en rappelant leur propriétés. La paragraphe 2.3.1.3 est consacré aux traces de ces sous-familles et résume en particulier les travaux effectués en collaboration avec Antoine Meyer et publiés dans [CM06a].

2.3.1.1 Transducteurs de mots

Les transducteurs de mots sont des automates finis acceptant les parties rationnels du monoïde produit $\Sigma^* \times \Sigma^*$. Un transducteur est simplement un automate fini sur ce monoïde. Un transducteur T sur un alphabet Σ est un automate fini étiqueté par $(\Sigma \cup \{\varepsilon\}) \times (\Sigma \cup \{\varepsilon\})$. Nous ne distinguerons pas le transducteur de la relation qu'il accepte et nous écrirons $(w,w') \in T$ si (w,w') est accepté par T. Pour une présentation détaillée, nous renvoyons le lecteur à [Ber79, Pri00, Sak03].

En général, il n'existe pas de borne sur la différence de la taille de l'entrée et de la sortie d'un transducteur. Des sous-classes pertinentes sont obtenues en imposant une forme de synchronisation entre l'entrée et la sortie du transducteur. C'est le cas des transducteurs lettre-à-lettre ou synchrone étiquetés par $\Sigma \times \Sigma$.

Une forme plus relaxée de synchronisation a été introduite par Elgot et Mezei dans [EM65]. Cette classe de relations a également été étudiée dans [FS93] sous le nom de *relations synchronisées*. Une relation est synchronisée à gauche si elle peut s'écrire comme une union finie de produits $R.S$, où R est une transduction lettre-à-lettre et S est de la forme (L,ε) ou (ε,L), avec L un langage régulier sur Σ. De façon équivalente, les relations synchronisées sont celles qui sont acceptées par les transducteurs dans lesquels pour tout chemin $q_0 \xrightarrow[q]{(x_0,y_0)}_1 \dots q_{n-1} \xrightarrow[q]{(x_n,y_n)}_n$, il existe $k \in [0,n]$ tel que pour tout $i \in [0,k-1]$, $x_i,y_i \in \Sigma$ et soit $x_k = \dots = x_n = \varepsilon$, soit $y_k = \dots = y_n = \varepsilon$. De tels transducteurs sont dits synchronisés à gauche. Les relations et les transducteurs synchronisés à droite sont définies de façon similaire.

La notion classique de déterminisme pour les automates n'a pas de sens dans le cas d'un monoïde qui n'est pas libre. La notion de transducteur séquentiel fournit une notion pertinente de déterminisme pour les transducteurs de mots. Nous dirons qu'un transducteur T est *séquentiel* si pour tous états $q,q',q'' \in Q$, si $q \xrightarrow{(x,y)} q'$ et $q \xrightarrow{(x',y')} q''$ alors soit $x = x'$, $y = y'$ et $q' = q''$, soit $x \neq \varepsilon$, $x' \neq \varepsilon$ et $x \neq x'$.

2.3.1.2 Définitions et propriétés

Dans ce paragraphe, nous présentons la famille générale des graphes rationnels et certaines de ses sous-familles, en particulier les graphes rationnels synchronisés (ou automatiques) et synchrones.

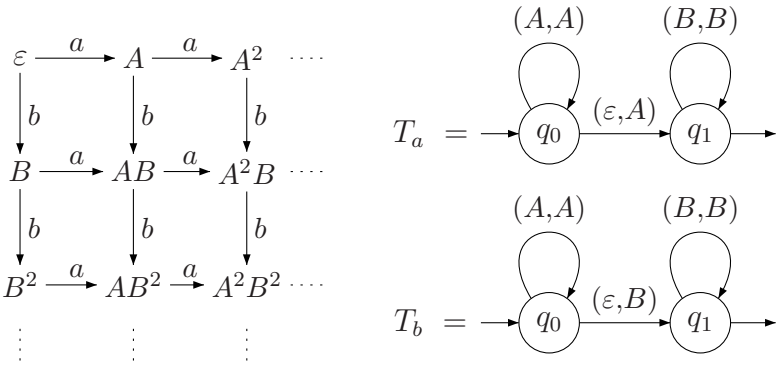


FIG. 2.9 – *La grille et les transducteurs la définissant.*

Les graphes rationnels sont simplement définis comme les graphes de calcul d'ensembles finis de transducteurs de mots. Ainsi, tout Σ-graphe rationnel G est caractérisé par une famille $(T_a)_{a\in\Sigma}$ de transducteurs de mots sur un alphabet Γ quelconque. Nous rappelons que par définition des graphes de calcul, il existe un arc étiqueté par a dans G entre deux sommets u et $v \in \Gamma^*$ si $(u,v) \in T_a$.

La figure 2.9 montre un exemple de graphe rationnel, la *grille* bi-dimensionnelle infinie, ainsi que les transducteurs la définissant.

Les graphes rationnels à transducteurs synchronisés ont été introduits dans [KN94, BG00] sous le nom de graphes *automatiques*, et par Rispal sous le nom de graphes rationnels synchronisés. Par un léger abus de langage, on parlera de graphes synchrones pour se référer à des graphes rationnels définis par des transducteurs synchrones, et de graphes séquentiels synchrones dans le cas correspondant.

Ces sous-familles forment une suite strictement croissante. Les graphes séquentiels synchrones sont strictement inclus dans les graphes synchrones car les graphes séquentiels synchrones sont nécessairement déterministes. Les graphes synchrones sont à leur tour strictement inclus dans les graphes synchronisés car les graphes synchrones ont un degré sortant fini alors que les graphes synchronisés peuvent avoir un degré sortant infini. L'inclusion stricte des graphes synchronisés dans les graphes rationnels est obtenue en considérant la croissance des degrés dans ces deux familles.

Proposition 2.3.1 ([Mor01])**.** *Pour tout graphe rationnel G de degré sortant fini et tout sommet x, il existe $c \in \mathbb{N}$ tel que le degré sortant de tout sommet à distance n de x est au plus c^{c^n}.*

Cette borne supérieure peut être atteinte : considérons le graphe rationnel non étiqueté $G_0 = \{T\}$ où T est le transducteur sur $\Gamma = \{A,B\}$ à un état q_0 à la fois

initial et final et une transition $q_0 \xrightarrow{(X,YZ)} q_0$ pour tous X,Y et $Z \in \Gamma$. Ce graphe a un degré sortant égal à $2^{2^{n+1}}$ à la distance n du sommet A. Dans le cas des graphes synchronisés de degré fini, la borne sur le degré sortant est simplement exponentielle.

Proposition 2.3.2 ([Ris02]). *Pour tout graphe synchronisé G de degré sortant fini et tout sommet x, il existe $c \in \mathbb{N}$ tel que le degré sortant de tout sommet à distance n de x est au plus c^n.*

Il suit de la proposition précédente que G_0 est rationnel mais n'est pas synchronisé.

Le graphe de l'exemple 2.9 n'ayant pas une théorie monadique décidable, les graphes rationnels n'ont donc pas une théorie monadique décidable. Dans [Mor01], l'auteur montre que la théorie au premier ordre des graphes rationnels est, elle aussi, indécidable. Une construction simplifiée est donnée dans [Tho01].

Théorème 2.3.3 ([Mor01]). *Il existe un graphe rationnel ayant une théorie au premier ordre indécidable.*

Une première manière d'obtenir un résultat de décidabilité est de restreindre l'expressivité des transducteurs définissant ces graphes. Ainsi les graphes synchronisés ont quant à eux tous une théorie au premier ordre décidable.

Théorème 2.3.4 ([BG00]). *Les graphes synchronisés ont une théorie au premier ordre décidable.*

Dans [CM06c] en collaboration avec Christophe Morvan, nous avons établi que la décidabilité de la théorie au premier ordre des graphes rationnels peut être obtenue non plus en restreignant l'expressivité des transducteurs mais en contraignant la structure des graphes. Nous avons établi que la théorie au premier ordre des arbres rationnels (*i.e.* des graphes rationnels qui sont des arbres) est décidable. De plus, nous avons montré que ce résultat ne peut être étendu ni en terme de structure ni en terme de logique. Nous avons construit un graphe rationnel orienté sans cycle ayant une théorie au premier ordre indécidable. De plus, nous avons construit un arbre rationnel ayant une théorie au premier ordre avec accessibilité rationnelle indécidable.

Théorème 2.3.5 ([CM06c]). *Les arbres rationnels ont une théorie au premier ordre décidable.*

2.3.1.3 Traces

Dans [MS01], Morvan et Stirling établissent que les traces des graphes rationnels d'un ensemble fini de sommets à un ensemble à fini de sommets sont les

langages contextuels. Cette propriété a été étendue aux graphes synchronisés par Rispal dans [Ris02]. Une version étendue de ces deux résultats a été publiée dans [MR05].

Les preuves fournies dans ces travaux reposent sur la forme normale de Penttonen des grammaires contextuelles [Pen74]. Cette forme normale présentent deux inconvénients majeurs: l'obtention de cette forme normale est loin d'être immédiate et de plus il n'existe pas de caractérisation des langages contextuels déterministes à partir des grammaires. En particulier, ces preuves ne nous permettent pas de définir une sous-famille des graphes rationnels traçant les langages contextuels déterministes.

Dans [CM06a], nous fournissons de nouvelles preuves de ces résultats basées sur la correspondance étroite entre les systèmes de pavages et les graphes synchrones. Les systèmes de pavages ont été définis à l'origine pour reconnaître des langages d'images (*i.e.* des tableaux finis de symboles sur un alphabet fini). Les langages d'images acceptés par ces systèmes sont aussi appelés des langages d'images locaux. Ils peuvent être vus comme des accepteurs de langages de mots en ne considérant que la première ligne de chaque image. Dans [LS97], les auteurs établissent que les ensemble des premières lignes des langage locaux d'images sont les langages contextuels.

Nous exploitons cette correspondance pour évaluer l'expressivité nécessaire en termes de transducteurs et en terme de structure du graphe (nombre de sommets initiaux, degré des sommets) pour qu'une sous-famille des graphes rationnels trace les langages contextuels.

Les résultats obtenus sont résumés dans le tableau 2.1. Chaque ligne du tableau correspond à une sous-famille des graphes rationnels. Chaque colonne correspond à une restriction structurelle portant sur le nombre de sommets initiaux et sur le degré des sommets du graphe. Dans la première colonne, nous considérons un ensemble rationnel quelconque de sommets initiaux et dans la seconde, nous considérons un ensemble rationnel de la forme i^*. Dans les deux dernières colonnes, nous considérons un unique sommet initial et le cas d'un sommet unique et du degré fini respectivement.

Une case contient un symbole d'équalité pour indiquer que les traces de la famille correspondante coïncident avec les langages contextuels. De même, un symbole d'inclusion indique que ses traces sont strictement incluses dans les langages contextuels. Une point d'interrogation dénote une conjecture.

Nous caractérisons précisément les traces des graphes synchronisés de degré fini à partir d'un unique sommet initial jusqu'à un ensemble rationnel de sommets finaux. Ces traces sont précisément les langages acceptés par les machines linéairement bornées effectuant au plus un nombre linéaire de changements de directions. Nous conjecturons que cette classe de langages est strictement incluse dans les langages contextuels. Cependant, il existe peu de résultats de séparation

	Ens. rat.	Ens. i^*	Som. unique	Som. unique (d° fini)
Rationnels [MS01]	$=$	$=$	$=$	$=$
Synchronisés [Ris02]	$=$	$=$	$=$	$\subset$ (?)
Synchrones [Ris02]	$=$	$=$	$\subset$	$\subset$
Séqu. synchrones	$=$	$\subset$ (?)	$\subset$	$\subset$

TAB. 2.1 – *Familles de graphes rationnels et leurs langages.*

pour les classes de complexité définies par des restrictions en temps et en espace (voir par exemple [vM04]). En particulier, les techniques de diagonalisation, utilisées pour montrer que la hiérarchie du temps polynomial est stricte (voir par exemple [For00]), ne peuvent être adaptées faute d'une notion pertinente de LBM universelle.

Enfin, nous définissons deux sous-familles des graphes rationnels dont les traces sont précisément les langages contextuels déterministes. Pour une présentation détaillée des ces résultats, nous renvoyons le lecteur à [CM06a].

2.3.2 Graphes linéairement bornés

Dans [CM05, CM06b] en collaboration avec Antoine Meyer, nous avons étudié la classe des graphes des transitions des machines linéairement bornées (cf. sous-paragraphe 1.2.3). Pour cette famille de systèmes de transitions étiquetées (LTS), les graphes des configurations sont simplement les graphes des LTS: il n'y a pas de restriction sur l'ensemble des configurations. Comme nous l'avons vu précédemment pour pouvoir définir les graphes des transitions, il nous faut considérer des LTS normalisés. Dans la suite, nous ne considérons que des LBM normalisées c'est-à-dire des LBM qui induisent des LTS normalisés au sens du sous-paragraphe 2.1.1.2. Il est aisé de vérifier que tout langage contextuel est accepté par une LBM normalisée.

Définition 2.3.6. Un graphe est dit *linéairement borné* s'il est isomorphe au graphe des transitions d'une machine linéairement bornée normalisée.

Une approche similaire a été proposée dans [KP99, Pay00]. Les auteurs définissent ce que nous appelons les graphes enracinés des LBM. Cependant ils ne donnent pas de notion de graphes des transitions (où les transitions étiquetées par τ n'apparaissent pas).

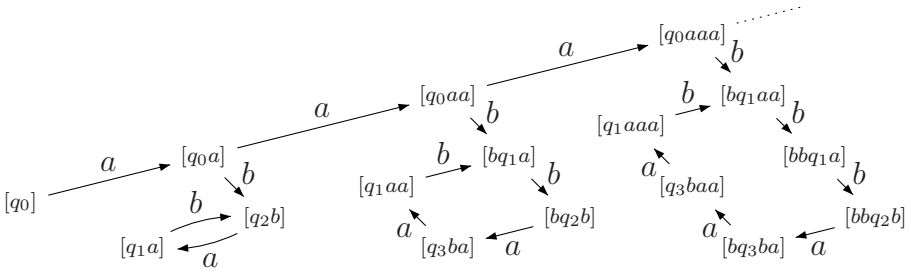


FIG. 2.10 – *Le graphe des transitions d'une LBM normalisée acceptant* $\{(a^n b^n)^+ \mid n \geq 1\}$.

Les graphes linéairement bornés sont clos par restriction aux sommets accessibles depuis un sommet donné et par restriction à un langage contextuel de configurations observables.

Exemple 2.3.7. La figure 2.10 présente le graphe des transitions d'une LBM M dont les transitions sont:

$$q_0] \xrightarrow[q_0]{a} a] \qquad q_1 a \xrightarrow[q_1]{b} b+ \qquad q_2 b \xrightarrow[q_3]{a} a- \qquad q_3 b \xrightarrow[q_3]{a} a-$$
$$q_0 a \xrightarrow[q_0]{a} aa \qquad q_1] \xrightarrow[q_2]{\varepsilon}]- \qquad q_3[\xrightarrow[q_1]{\varepsilon} [+$$
$$q_0 a \xrightarrow[q_1]{b} b+$$

et dont l'unique état final est q_2. Cette machine accepte le langage $\{(a^n b^n)^+ \mid n \geq 1\}$, qui est aussi la trace de son graphe des transitions entre le sommet $[q_0]$ et les sommets dans $[b^* q_2 b]$. Par soucis de clarté, nous ne présentons dans la figure 2.3.2 que la partie du graphe des transitions accessible depuis $[q_0]$. Comme nous l'avons déjà mentionné, les graphes linéairement bornés sont fermés par restriction aux sommets accessibles depuis un sommet donné. Le graphe présenté dans la figure 2.10 est donc un graphe linéairement borné.

Les traces des graphes linéairement bornés d'un ensemble fini de sommets à un ensemble fini de sommets sont les langages contextuels. En fait, on peut montrer que chaque langage contextuel est la trace d'un graphe linéairement borné déterministe. Ce résultat très fort est dû au fait que l'on considère des LBM qui ne terminent pas nécessairement leur calcul. Ces comportements divergents sont masqués par la τ-fermeture. Si l'on se restreint à des LBM qui terminent toujours, ce résultat n'est plus vrai. En fait, les traces des graphes de transitions déterministes des LBM qui terminent sont les langages contextuels déterministes. Ce résultat n'est pas immédiat car comme nous l'avons remarqué dans le paragraphe 2.1.1.2, une LBM non-déterministe peut avoir un graphe des transitions déterministe.

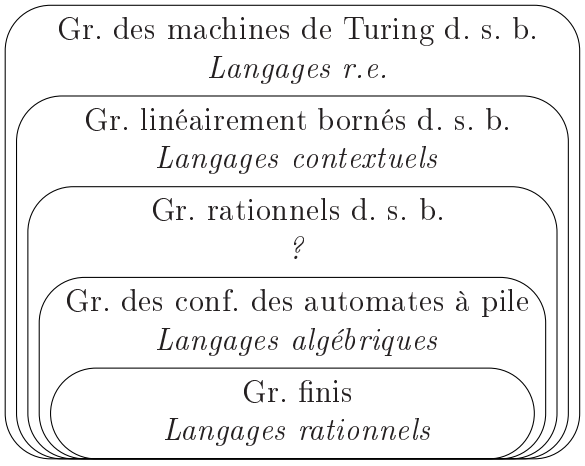


FIG. 2.11 – *Une hiérarchie à la Chomsky d'accepteurs infinis de degré sortant borné (d.s.b)*

Du point de vue de la logique, les graphes linéairement bornées ont une théorie au premier ordre indécidable. Cependant leur graphe des configurations ont une théorie au premier ordre décidable. En effet, ces graphes sont des graphes synchronisés (ou automatiques) (cf. théorème 2.3.4).

Cette remarque, nous amène à considérer le lien entre les graphes linéairement bornés et les graphes rationnels. Dans le cas général, ces deux familles sont incomparables. Ce résultat vient de la croissance des degrés dans les deux familles. Il est facile de vérifier que le degré sortant des graphes linéairement bornés croît de façon au plus exponentielle. Nous pouvons aisément construire un graphe linéairement borné dont le degré entrant croît de manière triplement exponentielle.

Cependant si l'on se restreint au cas où les graphes considérés ont un degré sortant borné, nous avons établi que les graphes rationnels sont strictement inclus dans les graphes linéairement bornés.

Théorème 2.3.8 ([CM05])**.** *Les graphes rationnels de degré borné sont des graphes linéairement bornés. De plus, il existe un graphe linéairement borné qui n'est isomorphe à aucun graphe rationnel.*

Ce résultat nous a amené à considérer une hiérarchie d'accepteurs différente de celle introduite dans [CK02] (cf. figure 2.1). Dans cette hiérarchie alternative qui est présentée dans la figure 2.11, nous ne considérons que des graphes de degré borné et les traces ne sont prises que par rapport à un unique sommet initial. Cette notion nous semblent plus proche de la notion intuitive d'automate.

Chapitre 3

Transformations de graphes

Une transformation T de graphes est une application qui associe à chaque graphe coloré G un graphe coloré $T(G)$. Cette transformation est dite *MSO-compatible* si pour toute formule monadique φ et pour tous ensembles finis Σ et C, nous pouvons effectivement calculer une formule φ^T telle que pour tout graphe sur (Σ,C):

$$T(G) \models \varphi \text{ si et seulement si } G \models \varphi^T$$

En particulier, si un graphe G a une théorie monadique décidable alors $T(G)$ a aussi une théorie monadique décidable.

Dans ce chapitre, nous présentons différentes transformations de graphes qui sont MSO-compatibles. Nous divisons ces transformations en trois catégories:

- les transformations définies par la logique monadique telles que les interprétations et les transductions monadiques (cf. paragraphe 3.1),
- les transformations définies à base d'automates finis qui sont toutes des cas particulier de transductions monadiques (cf. paragraphe 3.2),
- enfin, les transformations associant à chaque graphe G une structure arborescente telles que le dépliage ou le *treegraph* (cf. paragraphe 3.3).

À notre connaissance, il n'existe pas dans la littérature de transformation de graphe plus générale que celles présentées dans ce chapitre. Il est naturel de s'interroger sur la pertinence de l'étude des transformations basées sur des automates finis qui comme nous l'avons dit, sont des cas particuliers de transductions monadiques. Cette approche a été initiée par Caucal dans [Cau96] avec les substitutions rationnelles inverses. Leur intérêt est de donner une vision minimale de l'expressivité nécessaire à la réalisation d'une transformation donnée.

Dans le paragraphe 3.4, nous poursuivons cette approche et nous établissons des résultats de commutation partielle entre les interprétations monadiques et le dépliage. Ces résultats font apparaître de manière naturelle les transformations définies à base d'automates finis. Ces résultats ont été obtenus en collaboration avec Thomas Colcombet et publiés dans [CC03].

Enfin, nous concluons ce chapitre, par le paragraphe 3.5, en montrant que la transformation Treegraph préserve la propriété de sélection sur les arbres déterministes. Ce résultat a été formalisé lors d'une visite chez Alexander Rabinovich à l'université de Tel Aviv et financée par le programme Automatha de l'European Science Foundation.

3.1 Interprétations et transductions monadiques

Dans ce sous-paragraphe, nous présentons divers enrichissements des interprétations monadiques.

Une interprétation monadique $\mathcal{I}$ est donnée par un couple de familles de formules monadiques $((\varphi_a(x,y))_{a\in\Sigma},(\varphi_c(x)_{c\in C}))$ pour deux ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$. En appliquant $\mathcal{I}$ à un graphe G, nous obtenons le graphe sur (Σ,C):

$$\begin{array}{rcl} \mathcal{I}(G) & = & \{(u,a,v) \mid G \models \varphi_a[u,v], u,v \in V_G \text{ et } a \in \Sigma\} \\ & \cup & \{(c,u) \mid G \models \varphi_c[u], u \in V_G \text{ et } c \in C\}. \end{array}$$

Nous pouvons imposer que le graphe $\mathcal{I}(G)$ ne contienne pas de sommet isolé en remplaçant $\varphi_c(x)$ par $\varphi'_c(x) = \varphi_c(x) \wedge \exists y \bigvee_{a\in\Sigma}(\varphi_a(x,y) \vee \varphi_a(y,x))$. Nous supposerons toujours que les interprétations monadiques ne produisent pas de graphes possédant des sommets isolés.

Habituellement, la définition d'une interprétation monadique contient une formule supplémentaire $\delta(x)$ qui définit l'ensemble des sommets du graphe $\mathcal{I}(G)$. Dans notre présentation, ce comportement peut être obtenu en remplaçant les formules $\varphi_a(x,y)$ par $\varphi'_a(x,y) = \varphi_a(x,y) \wedge \delta(x) \wedge \delta(y)$ pour tout $a \in \Sigma$. Une interprétation monadique $\mathcal{I} = ((\varphi_a(x,y))_{a\in\Sigma},(\varphi_c(x)_{c\in C}))$ avec $\varphi_a(x,y) = E_a(x,y) \wedge \delta(x) \wedge \delta(y)$ et $\varphi_c(x) = P_c x \wedge \exists y \bigvee_{a\in\Sigma}(E_a(x,y) \vee E_a(y,x)) \wedge \delta(x) \wedge \delta(y)$ pour une certaine formule $\delta(x)$ sera appelée une *restriction monadique* et sera uniquement données par les deux ensembles finis Σ and C et avec la formule $\delta(x)$.

Si une interprétation $\mathcal{I} = ((\varphi_a(x,y))_{a\in\Sigma},(\varphi_c(x)_{c\in C}))$ ne modifie pas la structure des graphes sur (Σ,C) mais seulement les couleurs (*i.e.* pour tout $a \in \Sigma$, $\varphi_a(x,y) = E_a(x,y)$), nous parlerons de *coloriage monadique* et nous omettrons les formules φ_a dans leur définition.

Une forme plus générale d'interprétation apparaît dans la littérature. Elle inclut un quotient par une relation d'équivalence MSO-définissable. Une *interprétation monadique avec quotient* $\mathcal{I}_\varepsilon$ est donnée par un uplet de formules monadiques $((\varphi_a(x,y))_{a\in\Sigma}, (\varphi_c(x)_{c\in C}), \varepsilon(x,y))$ pour deux ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$. Pour tout graphe coloré G, nous noterons $\mathcal{E}_G \subseteq V_G \times V_G$ la plus petite relation d'équivalence contenant la relation $\{(u,v) \mid \varepsilon[u,v]\}$ induite par la formule $\varepsilon(x,y)$. Remarquons que la relation $\mathcal{E}_G$ est définissable dans MSO. Il suffit de considérer la formule $\varphi(x,y) = \exists X_0\, \psi(X_0,x) \wedge (\forall X\ (\psi(X,x) \wedge X \subseteq X_0) \rightarrow X = X_0) \wedge y \in X_0$

où $\psi(X,x) = x \in X \wedge \forall y,z,\, [y \in X \wedge (\varepsilon(y,z) \vee \varepsilon(z,y)) \rightarrow z \in X]$. Pour tout sommet $u \in V_G$, nous noterons $[u]_{\mathcal{E}} = \{v \mid (u,v) \in \mathcal{E}_G\}$ la classe d'équivalence du sommet u pour $\mathcal{E}_G$. En appliquant $\mathcal{I}_\varepsilon$ à un graphe G, nous obtenons le graphe:

$$\begin{aligned} \mathcal{I}_\varepsilon(G) \;\; = \;\; & \{([u]_{\mathcal{E}},a,[v]_{\mathcal{E}}) \mid G \models \varphi_a[u,v], u,v \in V_G \text{ et } a \in \Sigma\} \\ & \cup \{(c,[u]_{\mathcal{E}}) \mid G \models \varphi_c[u] \text{ et } u \in V_G\}. \end{aligned}$$

Comme pour les interprétations monadiques, nous pouvons imposer syntactiquement que $\mathcal{I}_\varepsilon(G)$ ne contienne pas de sommet isolé. Et nous supposerons que c'est toujours le cas.

Dans la théorie des modèles finis, des interprétations appelées interprétations multidimensionnelles sont considérées (voir par exemple [EF95]). Elles permettent en particulier de définir le produit synchronisé avec un graphe fini. Pour les graphes, ces interprétations multidimensionnelles sont appelées des transductions monadiques. Nous les présentons comme la composition d'une opération de copie par un ensemble fini et d'une interprétation monadique (à une seule dimension) en suivant [Cou94].

Pour tout ensemble fini $K \subset \Lambda$, l'opération de copie par K est appliquée à un graphe G pour obtenir le graphe:

$$K(G) \;\; = \;\; G \cup \{(u,k,u_k) \mid u \in V_G, k \in K, u_k \in V_k\}$$

où les ensembles $(V_k)_{k \in K}$ sont deux à deux disjoints et sont aussi disjoints de l'ensemble V_G mais sont en bijection avec V_G. Pour tout $k \in K$ et pour tout $u \in V_G$, nous notons u_k l'élément de V_k correspondant à u. Intuitivement, l'opération de copie par K crée $|K|$ copies distinctes de chaque sommet de G.

Une *transduction monadique* $\mathcal{T}$ est définie par un couple $(K,\mathcal{I})$ où $K \subset \Lambda$ est un ensemble fini et où $\mathcal{I}$ est une interprétation monadique. À chaque graphe G, la transduction monadique $\mathcal{T}$ associe le graphe $\mathcal{T}(G) = \mathcal{I}(K(G))$. Nous dirons qu'un graphe H est MSO-définissable dans un graphe G s'il existe une transduction monadique $\mathcal{T}$ telle que $H \approx \mathcal{T}(G)$.

Exemple 3.1.1. Considérons la transduction monadique $\mathcal{T}_0$ définie comme la composition de la copie par l'ensemble $\{b\}$ et de l'interprétation $\mathcal{I}_0 = (\varphi_x)_{x \in \{a,b\}}$ où $\varphi_a(x,y) = E_a(x,y)$ et $\varphi_b(x,y) = (\exists z_0,z_1, E_b(z_0,x) \wedge E_a(z_1,z_0) \wedge E_b(z_1,y)) \vee E_b(x,y)$. La figure 3.1 présente le résultat de l'application de $\mathcal{T}_0$ à la demi-droite Δ_1.

La propriété fondamentale des transductions monadiques est qu'elles sont MSO-compatibles. La proposition suivante résume leurs propriétés de bases.

Proposition 3.1.2. *Les interprétations monadiques avec ou sans quotient et les transductions monadiques sont MSO-compatibles. Les classes des interprétations, des coloriages, des restrictions, des interprétations avec quotient et des transductions monadiques sont fermées par composition.*

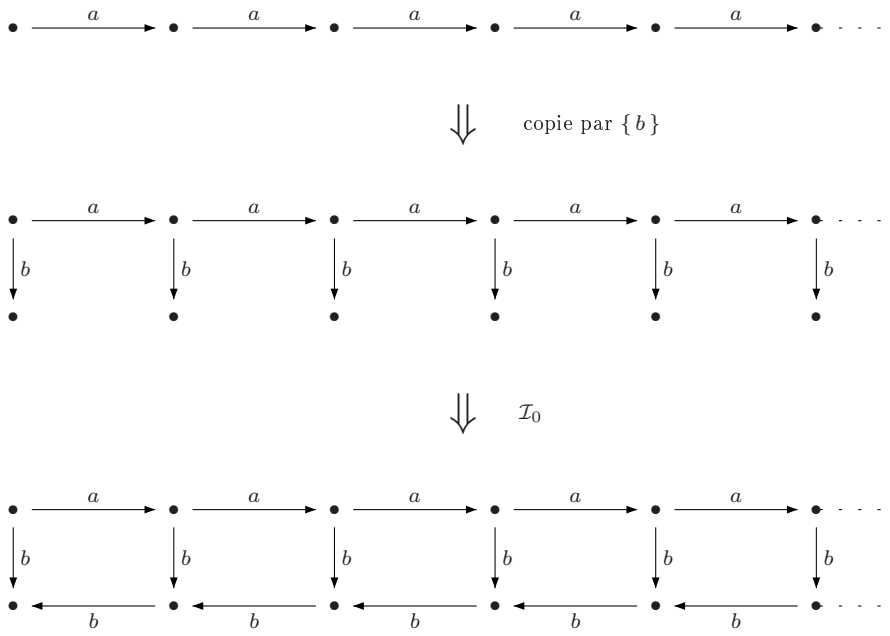


FIG. 3.1 – *La transduction $\mathcal{T}_0$ à la demi-droite Δ_1.*

La notion de MSO-compatibilité pour les transformations de graphes ne s'intéresse qu'aux formules sans variables libres. Dans le cas des transductions monadiques, cette propriété s'étend aux formules avec variables libres.

Proposition 3.1.3. *Pour toute transduction $\mathcal{T}$ et pour toute formule monadique $\varphi(X_1,\ldots,X_n)$, nous pouvons effectivement calculer une formule $\varphi^{\mathcal{T}}(X_1,\ldots,X_n)$ telle que pour tout graphe G et tous ensembles $U_1,\ldots,U_n \subseteq V_G \cap V_{\mathcal{T}(G)}$, nous avons:*

$$\mathcal{T}(G) \models \varphi[U_1,\ldots,U_n] \Leftrightarrow G \models \varphi^{\mathcal{T}}[U_1,\ldots,U_n]$$

Pour les arbres déterministes, les interprétations monadiques avec quotient peuvent être remplacées par des transductions monadiques.

Proposition 3.1.4. *Pour toute interprétation monadique avec quotient $\mathcal{I}_\varepsilon$, il existe une transduction $\mathcal{T}$ telle que pour tout arbre déterministe t, $\mathcal{I}_\varepsilon(t) \approx \mathcal{T}(t)$.*

Démonstration. Soit $\mathcal{I}_\varepsilon = ((\varphi_a(x,y))_{a\in\Gamma},(\varphi_c(x))_{c\in C'},\varepsilon(x,y))$ une interprétation avec quotient et soit Σ (resp. C) l'ensemble des étiquettes (resp. couleurs) apparaissant dans les formules définissant $\mathcal{I}$. Comme nous l'avons déjà remarqué, il existe une formule monadique $\varphi_\varepsilon(x_0,x_1)$ telle que pour tout graphe G, la relation

$E_\varepsilon^G = \{(u,v) \in V_G \times V_G \mid G \models \varphi_\varepsilon[u,v]\}$ est la plus petite relation d'équivalence contenant $\{(u,v) \in V_G \times V_G \mid \varepsilon[u,v]\}$. Nous noterons $[u]_\varepsilon$ la classe d'équivalence u de E_ε^G. Soit $A = (Q \times 2^{\{0,1\}} \times 2^{\{0,1\}}, I, \Delta, \Omega)$ un automate d'arbres à parité acceptant $\varphi_\varepsilon(x_0,x_1)$ normalisé comme décrit dans le paragraphe 1.4.3.

L'idée de la preuve est de donner une caractérisation unique de chaque classe d'équivalence de E_ε^t par un sommet de t et par une quantité d'information finie. Dans ce but, pour tout arbre t sur (Σ,C), nous associons à chaque classe d'équivalence $X \subseteq \mathrm{Dom}(t)$ de E_ε^t le couple (w_X,Δ_X) où $w_X \in \Sigma^*$ est le plus petit préfixe commun de l'ensemble X, et $\Delta_X \subseteq \Delta$ est l'ensemble des transitions δ pour lesquelles il existe une exécution acceptante ρ de A sur t avec $\Phi_\rho(w_X) = \delta$ acceptant un couple (x,x') d'éléments de X.

Fait 1 Pour toutes classes d'équivalence X et Y de E_ε^t, si $(w_X,\Delta_X) = (w_Y,\Delta_Y)$ alors $X = Y$.

Supposons que $w_X = w_Y = w$ et $\Delta_X = \Delta_Y = D$ et supposons par l'absurde que X et Y ne soient pas égales. Comme X et Y sont des classes d'équivalence, X et Y sont disjointes. Nous distinguons deux cas.

Cas $w \in X$**.** Il existe une exécution acceptante ρ de A pour le couple (w,w). La transition $\delta = \varphi_{\rho(w)}$ est de la forme $(q,\{0,1\},\{0,1\}),c,f)$ et appartient par définition à $D = \Delta_Y$. Par définition de Δ_Y, il existe une exécution acceptante de A avec $\Phi_\rho = \delta$ accepte un couple d'éléments de Y. D'après la forme de δ, ρ accepte (w,w) et donc w appartient à Y ce qui amène la contradiction.

Cas $w \notin X$**.** Par définition de w, il existe deux éléments de x et y dans X tels que $x = waw_x$ et $y = wbw_y$ avec $a \neq b \in \Sigma$ et $w_x,w_y \in \Sigma^*$. Il existe une exécution ρ de A acceptant le couple (x,y). La transition $\delta = \Phi_\rho(w)$ est de la forme $((q,\emptyset,\{0,1\}),c,f)$ où $f(a) = (q_a,X_a,\{0\})$ et $f(b) = (q_b,X_b,\{1\})$. Comme δ appartient à Δ_Y, il existe une exécution ρ' de A acceptant un couple (x',y') d'éléments de Y. Par définition de δ, nous avons $x' = xaw'_x$ et $y' = ybw'_y$.

Nous pouvons construire une exécution σ de A acceptant le couple (x,y'). Nous prenons simplement $\sigma(v) = \rho(v)$ si wb n'est pas un préfixe de v et $\sigma(v) = \rho'(v)$ sinon. L'exécution σ est bien formée car $\rho(wb) = \rho'(wb)$. Elle est acceptante car ρ et ρ' le sont et par construction, elle accepte le couple (x,y') Donc $(x,y') \in E_\varepsilon^t$ et $y' \in X$, ce qui amène la contradiction.

Fait 2 Pour tout sous-ensemble $D \subseteq \Delta$, il existe une formule monadique $\varphi_D(x,X)$ telle que pour tout arbre déterministe t sur (Σ,C) et pour tout $u \in \mathrm{Dom}(t)$ et $U \subseteq \mathrm{Dom}(t)$, $t \models \varphi_D(u,U)$ si et seulement si U est une classe d'équivalence de E_ε^t et $u = w_U$ et $D = \Delta_U$.

Considérons la formule $\psi(x,X)$ qui exprime que x est le plus petit ancêtre commun des éléments de X, et la formule $\xi(X)$ qui exprime que X est une classe d'équivalence de $\varepsilon(x,y)$. Pour tout $\delta \in \Delta$, considérons la formule $\varphi_\delta(x,X)$ qui affirme qu'il existe une exécution de A acceptant un couple d'éléments de X en

utilisant la transition δ sur le nœud x.

Nous pouvons supposer sans perte de généralité que 2^Δ est disjoint de Σ et de C. La transduction $\mathcal{T}$ est donnée par $(2^\Delta,\mathcal{I})$. L'interprétation $\mathcal{I}$ est définie par $(\psi_a(x,y))_{a\in\Sigma}$ et $(\psi_c(x))_{c\in C'}$ où

$$\begin{array}{lcl}\psi_a(x,y) & = & \exists X,Y,x_0,y_0,\\ & & \bigvee_{D,D'\subseteq\Delta} E_D(x_0,x)\wedge E_{D'}(y_0,y)\wedge\varphi_D(x_0,X)\wedge\varphi_{D'}(y_0,Y)\\ & & \wedge(\exists x',y',\, x'\in X\wedge y'\in Y\wedge\varphi_a(x',y'))\\ \psi_c(x) & = & \exists X,x_0,\ \bigvee_{D\subseteq\Delta} E_D(x_0,x)\wedge\varphi_D(x_0,X)\wedge(\exists x',\, x'\in X\wedge\varphi_c(x'))\end{array}$$

Il suit des faits 1 et 2 que pour tout arbre déterministe t sur (Σ,C), $\mathcal{T}(t)\approx\mathcal{I}_\varepsilon(t)$. □

3.2 Transformations à base d'automates finis

Dans ce paragraphe, nous présentons diverses transformations de graphes basées sur des automates finis.

Un coloriage de graphes R est donné par un sous-ensemble fini de $\Theta\times\Theta$. En appliquant R à un graphe G, nous obtenons le graphe:

$$\begin{array}{lcl}R(G) & = & G\ \cap\ V_G\times\Lambda\times V_G\\ & \cup & \{(d,u)\mid(c,u)\in G\text{ et }(c,d)\in R\}.\end{array}$$

Intuitivement pour toute paire $(c,d)\in R$, tous les sommets coloriés par c dans G sont recoloriés par d dans $R(G)$.

Un coloriage rationnel μ est donné par une application de Θ dans $\mathrm{Rat}((\Lambda\cup\Theta)^*)$ de support fini (*i.e.* l'ensemble $\{c\in\Theta\mid\mu(c)\neq\emptyset\}$ est fini). Lorsque l'on applique le recoloriage rationnel μ à un graphe G à partir d'un sommet $s\in V_G$, nous obtenons le graphe:

$$\begin{array}{lcl}\mu_s(G) & = & G\ \cap\ V_G\times\Lambda\times V_G\\ & \cup & \{(d,u)\mid s\xrightarrow[G]{w}u,\, d\in\Theta\text{ et }w\in\mu(d)\}.\end{array}$$

Comme l'accessibilité par un chemin dont l'étiquette appartient à un langage rationnel donné $R\in\mathrm{Rat}((\Lambda\cup\Theta)^*)$ est exprimable en logique monadique, le coloriage rationnel est un cas particulier de coloriage monadique.

Une substitution rationnelle inverse h [Cau92] est donnée par une application de Λ dans $\mathrm{Rat}((\Lambda\cup\overline{\Lambda}\cup\Theta)^*)$ de support fini. Elle est appliquée à un graphe G pour donner le graphe:

$$\begin{array}{lcl}h^{-1}(G) & = & \{(u,a,v)\mid u\xrightarrow[G]{w}v,\, u,v\in V_G,\, w\in h(a)\text{ et }a\in\Gamma\}\\ & & \cup\,\{(c,u)\mid u\in V_{h^{-1}(G)}\text{ et }(c,u)\in G\}.\end{array}$$

Si tous les ensembles apparaissant dans $\mathrm{Im}(h)$ sont finis, nous parlerons de substitution finie.

Une *substitution rationnelle inverse colorée* g est donnée par un couple (R,h) où R est un recoloriage de graphe et h est une application rationnelle. Elle est appliquée par inverse à un graphe G pour donner le graphe, noté $g^{-1}(G)$, égal à $R(h^{-1}(G))$.

Les substitutions rationnelles inverses et leurs variantes colorées sont des cas particuliers d'interprétations monadiques.

Sur les arbres déterministes, l'interprétation monadique peut être remplacée par deux transformations plus simples: le coloriage monadique et la substitution rationnelle inverse.

Proposition 3.2.1. *Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$ et pour toute interprétation monadique $\mathcal{I}$, il existe un coloriage monadique $\mathcal{M}$ et une substitution rationnelle inverse h tels que pour tout arbre déterministe t sur (Σ,C),*

$$\mathcal{I}(t) = h^{-1}(\mathcal{M}(t))$$

Démonstration. Soient Σ,Γ des sous-ensembles finis de Λ et soient C,D des sous-ensembles finis de Θ et $\mathcal{I} = ((\varphi_a(x,y))_{a\in\Gamma},(\varphi_c(x))_{c\in D})$ est une interprétation monadique.

Pour tout $a \in \Gamma$, il existe un automate d'arbres à parité normalisé $A_a = (Q_a \times 2^{\{0,1\}} \times 2^{\{0,1\}},I_a,\Delta_a,\Omega_a)$ acceptant $\varphi_a(x,y)$ (cf paragraphe 1.4.4). Nous pouvons supposer sans perte de généralité que $\Delta_a \cap \Delta_{a'} = \emptyset$ pour tout $a \neq a' \in \Sigma$.

Soit $\widetilde{\Delta}$ un sous-ensemble de Θ disjoint de C et en bijection avec $\Delta = \bigcup_{a\in\Sigma} \Delta_a$. Pour tout $\delta \in \Delta$, nous notons $\tilde{\delta}$ le symbole correspondant de $\tilde{\Delta}$. Pour tout $\delta \in \Delta_a$, le coloriage monadique $\mathcal{M}$ ajoute la couleur $\overline{\delta}$ au nœud $u \in \mathrm{Dom}(t)$ s'il existe une exécution ρ de A_a sur t avec $\Phi_\rho(u) = \delta$. Pour tout $a \in \Sigma$ et pour tout $\delta \in \Delta_a$, notons $\varphi_{\tilde{\delta}}(x)$ la formule monadique telle que pour tout arbre déterministe t sur (Σ,C) et pour tout $u \in \mathrm{Dom}(t)$, $t \models \varphi_{\tilde{\delta}}[u]$ si et seulement si il existe une exécution acceptante ρ de A_a sur t avec $\Phi_\rho(u) = \delta$. Le coloriage monadique $\mathcal{M}$ est défini par $(\varphi_{\tilde{\delta}}(x))_{\delta\in\Delta} \cup (\varphi_d(x))_{d\in D}$.

Pour tout $b \in \Gamma$, nous définissons un langage rationnel $R_b \subseteq (\Sigma \cup \bar{\Sigma} \cup \tilde{\Delta})^*$ tel que pour tout arbre déterministe t sur (Σ,C), $u \xrightarrow[\mathcal{M}(t)]{w} v$ pour $w \in R_b$ si et seulement si $t \models \varphi_b[u,v]$.

Pour tout $a \in \Sigma$, $b \in \Gamma$ et tout $\delta \in \Delta_a$, nous définissons deux langages rationnels $M^b_{\delta,a}$ et $N^b_{\delta,a}$ inclus dans $(\Sigma \cup \overline{\Sigma} \cup \widetilde{\Delta})^*$. Pour tout $\delta \in \Delta_a$, nous prenons $\delta = (q_\delta,C_\delta,f_\delta)$ avec $q_\delta = (p_\delta,M_\delta,N_\delta)$.
Nous définissons $M^b_{\delta,a}$ comme étant l'ensemble des mots de la forme $\tilde{\delta}_0\overline{a}_0 \ldots \overline{a}_n\tilde{\delta}_{n+1}$ dans $\Delta_b(\overline{\Sigma}\Delta_b)^*$ avec $n \geq 0$ et tel que pour tout $i \in [1,n+1]$, $f_{\delta_i}(a_{i-1}) = q_{\delta_{i-1}}$, $M_{\delta_0} = \{0\}$, $N_{\delta_0} = \{0\}$, $M_{\delta_i} = \emptyset$, $N_{\delta_i} = \{0\}$ for $i \in [1,n]$ et $N_{\delta_{n+1}} = \{0,1\}$, $\delta = \delta_{n+1}$ et $a = a_n$.

Nous définissons $N^b_{\delta,a}$ comme l'ensemble des mots $\tilde{\delta}_0 a_0 \ldots a_n \tilde{\delta}_{n+1}$ dans $\Delta_b(\Sigma\Delta_b)^*$ avec $n \geq 0$ tels que pour tout $i \in [0,n]$, $f_{\delta_i}(a_{i-1}) = q_{\delta_{i+1}}$, $N_{\delta_0} = \{0,1\}$, $\delta = \delta_0$ and $a = a_0$, $M_{\delta_i} = \emptyset$, $N_{\delta_i} = \{1\}$ pour tout $i \in [1,n]$ et $M_{\delta_{n+1}} = \{1\}$ et $N_{\delta_{n+1}}$.
Pour tout $b \in \Gamma$, nous définissons le langage rationnel R_b de mots sur $(\Sigma \cup \bar{\Sigma} \cup \widetilde{\Delta})$ par:

$$\begin{array}{rcl} R_b & = & \bigcup_{\{\delta \mid M_\delta = \emptyset\}} \bigcup_{a \neq a' \in \Sigma} M^b_{\delta,a} N^b_{\delta,a'} \\ & \cup & \bigcup_{\{\delta \mid M_\delta = \{1\}\}} \bigcup_{a \in \Sigma} M^b_{\delta,a} \\ & \cup & \bigcup_{\{\delta \mid M_\delta = \{0\}\}} \bigcup_{a \in \Sigma} N^b_{\delta,a} \\ & \cup & \bigcup_{\{\delta \mid M_\delta = \{0,1\}\}} \tilde{\delta} \end{array}$$

Par une récurrence immédiate sur la longueur de w, nous établissons que:

- Pour tout $b \in \Gamma$ et pour tout $\delta \in \Delta_b$ telle que $M_\delta = \emptyset$, si $u \xrightarrow[\mathcal{M}(t)]{w} v$ pour un certain $w \in \bigcup_{a \neq a' \in \Sigma} M^b_{\delta,a} N^b_{\delta,a'}$ alors $t \models \varphi_b[u,v]$.
- Pour tout $b \in \Gamma$ et pour tout $\delta \in \Delta_b$ telle que $M_\delta = \{1\}$, si $u \xrightarrow[\mathcal{M}(t)]{w} v$ pour un certain $w \in \bigcup_{a \in \Sigma} M^b_{\delta,a}$ alors $t \models \varphi_b[u,v]$.
- Pour tout $b \in \Gamma$ et pour tout $\delta \in \Delta_b$ telle que $M_\delta = \{0\}$, $u \xrightarrow[\mathcal{M}(t)]{w} v$ pour un certain $w \in \bigcup_{a \in \Sigma} N^b_{\delta,a}$ alors $t \models \varphi_b[u,v]$.

Il s'en suit alors que pour tout arbre déterministe t sur (Σ,C), si $u \xrightarrow[\mathcal{M}(t)]{w} v$ pour un certain $w \in R_b$ alors $t \models \varphi_b[u,v]$. Réciproquement pour tout arbre déterministe t sur (Σ,C), si $t \models \varphi_b[u,v]$ alors $u \xrightarrow[\mathcal{M}(t)]{w} v$ pour $w \in R_b$.

Considérons la substitution rationnelle inverse colorée g donnée par (h,R) où h est définie pour tout $b \in \Gamma$ par $h(b) = R_b$ et où R est égal à $\{(d,d) \mid d \in D\}$. D'après les propriétés des langages R_b et pour tout arbre déterministe t sur (Σ,C), nous avons:

$$\mathcal{I}(t) = g^{-1}(\mathcal{M}(t)).$$

□

3.3 Dépliage et Treegraph

Le dernier type de transformations MSO-compatibles associe à un graphe G un arbre ou une structure arborescente basée sur G. Une transformation de ce type et qui a attiré beaucoup d'attention est l'itération de structure. La première mention en est faite dans [She75] dans une version plus faible (qui ne fait pas intervenir le prédicat de clone cl défini ci-après). La version que nous présentons est due à Muchnick et apparaît pour la première fois dans [Sem84].

Pour toute structure relationnelle $\mathcal{R} = (U,(R^{\mathcal{R}})_{R\in\mathcal{S}})$ sur une signature $\mathcal{S}$ qui ne contient pas cl et son, l'itération de $\mathcal{R}$ est la structure notée $\mathcal{R}^* = (U^*,(R^{\mathcal{R}^*})_{R\in\mathcal{S}},\mathrm{son}^{\mathcal{R}*},\mathrm{cl}^{\mathcal{R}^*})$ où l'univers U^* est l'ensemble des suites finies d'éléments de U et où:

- $R^{\mathcal{R}^*} = \{(wu_1,\ldots,wu_{|R|}) \mid w \in U^* \text{ et } (u_1,\ldots,u_{|R|}) \in R^{\mathcal{R}}\}$ pour tout $R \in \mathcal{S}$,
- $\mathrm{son}^{\mathcal{R}^*} = \{(w,wu) \mid u \in U,\, w \in U^*\}$ et où $\mathrm{cl}^{\mathcal{R}^*} = \{wuu \mid u \in U,\, w \in U^*\}$.

Remarquons que l'itération d'un graphe infini de degré sortant borné a un degré sortant infini à cause de la relation $\mathrm{son}^{\mathcal{R}^*}$. De plus, cette opération transforme un graphe non-coloré en un graphe coloré à cause du prédicat cl. C'est pourquoi nous présentons une variante due à Caucal qui n'introduit pas de couleur supplémentaire ni de degré sortant infini et qui est donc plus adaptée à notre étude.

Le Treegraph d'un graphe G par un symbole $\sharp \in \Lambda - \Lambda_G$ est le graphe $\mathrm{Treegraph}(G,\sharp)$ dont les sommets sont les suites finies de sommets de G et tel que:

$$\begin{aligned}\mathrm{Treegraph}(G,\sharp) \;=\; & \{(wu,a,wv) \mid w \in V_G^*,u,v \in V_G \text{ et } (u,a,v) \in G\}\\ & \cup\{(wu,\sharp,wuu) \mid w \in v_G^* \text{ et } u \in V_G\}\\ & \cup\{(c,wu) \mid w \in V_G^* \text{ et } (c,u) \in G\}\end{aligned}$$

Comme $\mathrm{Treegraph}(G,\sharp) \models E_\sharp xy[u,v] \Leftrightarrow G^* \models \mathrm{son}(x,y)\wedge\mathrm{cl}(y)[u,v]$, l'opération de Treegraph est toujours définissable dans l'itération G^*. Réciproquement, si G est connecté, nous pouvons définir G^* dans $\mathrm{Treegraph}(G,\sharp)$ en utilisant les équivalences suivantes:

$$\begin{aligned}G^* \models \mathrm{cl}(x)[u] &\Leftrightarrow \mathrm{Treegraph}(G,\sharp) \models \exists z E_\sharp zx[u],\\ G^* \models \mathrm{son}(x,y)[u,v] &\Leftrightarrow \mathrm{Treegraph}(G,\sharp) \models \exists z E_\sharp zy \wedge \mathrm{Reach}_{\Sigma\cup\bar{\Sigma}}(z,y)[u,v].\end{aligned}$$

La forme monadique $\mathrm{Reach}_{\Sigma\cup\bar{\Sigma}}(z,x)$ exprime l'existence d'un chemin de z à y dans lequel les arcs peuvent être pris dans les deux sens.

Une opération plus naturelle que le Treegraph est l'opération de dépliage. Le *dépliage* d'un graphe G d'un sommet $r \in V_G$, noté $\mathrm{Unf}(G,r)$, est l'arbre défini par:

$$\begin{aligned}& \{(\pi,a,\pi au) \mid \pi,\pi au \text{ des chemins dans } G \text{ commençants en } r\}\\ \cup\; & \{(c,\pi) \mid \pi \text{ est un chemin dans } G \text{ de } r \text{ à } v \text{ et } (c,v) \in G\}\end{aligned}$$

Courcelle et Walukiewicz ont montré dans [CW98] que le dépliage $\mathrm{Unf}(G,r)$ à partir d'un sommet r définissable en MSO est définissable dans l'itération G^*.

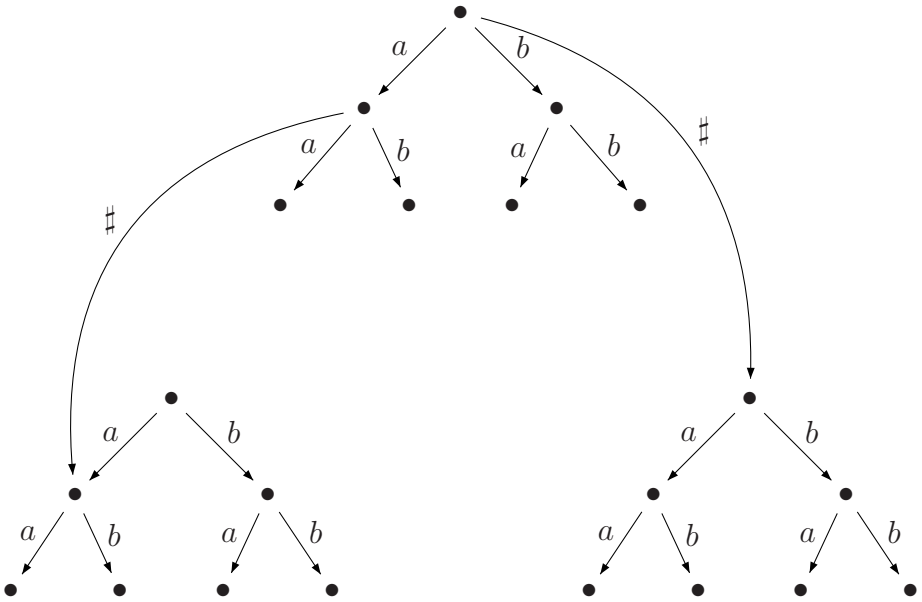


FIG. 3.2 – *Le Treegraph de l'arbre binaire* Δ_2 *par le symbole* $\sharp$.

Comme celle la composante connexe contenant r importe pour définir le dépliage et que r est définissable, Unf(G,r) est aussi définissable dans Treegraph(G,$\sharp$).

Exemple 3.3.1. Considérons le graphe G obtenu dans l'exemple 3.1.1 après l'application de la transduction $\mathcal{T}_0$ à la demi-droite Δ_1. La figure 3.3 présente son dépliage à partir de l'unique sommet G qui n'est destination d'aucun arc.

Pour les arbres déterministes, nous pouvons établir le résultat réciproque. Il est possible de définir le Treegraph de G par un symbole $\sharp$ en utilisant l'opération de dépliage et des interprétations monadiques. Cette construction est due à Colcombet et est présentée dans [Col04].

Lemme 3.3.2. *Pour tous ensembles finis* $\Sigma \subset \Lambda$ *et* $C \subset \Theta$ *et pour tout symbole* $\sharp \in \Lambda - \Sigma$*, il existe deux substitutions rationnels inverses* h_1 *et* h_2 *et une restriction monadique* $\mathcal{R}$ *telle que pour tout arbre déterministe* t *sur* (Σ,C)*,*

$$\text{Treegraph}(t,\sharp) \approx h_2^{-1}(\mathcal{R}(\text{Unf}(h_1^{-1}(t),\varepsilon))).$$

Démonstration. Soient $\Sigma \subset \Lambda$ et $C \subset \Theta$ deux ensembles finis et soit $\sharp \in \Lambda - \Lambda_G$ une étiquette. Prenons $\tilde{\Sigma} \subset \Lambda$ un ensemble d'étiquettes disjoint de Σ mais en bijection avec ce dernier. La substitution rationnelle inverse h_1 ajoute les arcs

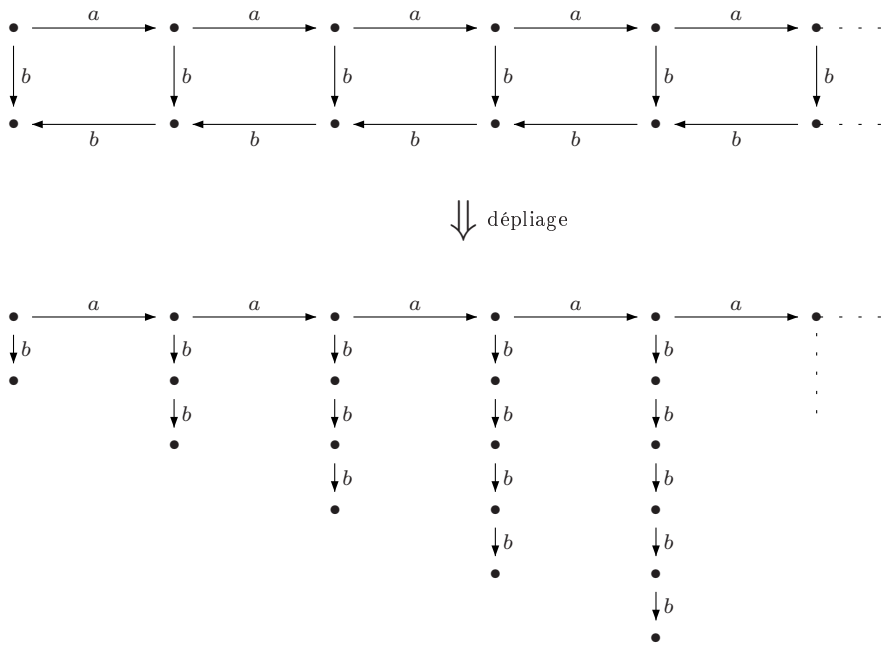


FIG. 3.3 – *Exemple de dépliage d'un graphe.*

retours étiqueté par les éléments correspondants de $\tilde{\Sigma}$ et des boucles étiquetées par $\sharp$ sur chaque sommet. Formellement $h_1(a) := \{a\}$, $h_1(\tilde{a}) := \{\bar{a}\}$ pour tout $a \in \Sigma$ et $h_1(\sharp) := \{\epsilon\}$. Ainsi h_1 préserve le déterminisme de t. La restriction monadique $\mathcal{R}$ ne conserve que les sommets v de $\mathrm{Unf}(h_1^{-1}(T),r)$ dont l'unique chemin depuis la racine ε ne contient pas de facteurs de la forme $a\tilde{a}$ ou $\tilde{a}a$ avec $a \in \Sigma$. Enfin, la substitution rationnelle inverse renverse les arcs étiquetés par une étiquette dans $\tilde{\Sigma}$. Formellement h_2 est donnée par $h_2(\sharp) = \{\sharp\}$ et $h_2(a) = \{a\} \cup \{\widetilde{\bar{a}}\}$ pour tout $a \in \Sigma$. □

Exemple 3.3.3. La figure 3.4 illustre cette construction. Le graphe du haut représente la demi-droite après l'application de h_1^{-1}. Le graphe du bas représente son dépliage. Les sommets pleins sont ceux conservés par la restriction monadique $\mathcal{R}$.

Remarquons que l'énoncé du lemme précédent est uniforme au sens où les substitutions rationnelles inverses et la restriction monadique ne dépendent que de l'ensemble des étiquettes de l'arbre et pas de l'arbre lui-même.

Le théorème de Muchnick, qui apparaît pour la première fois dans [Sem84],

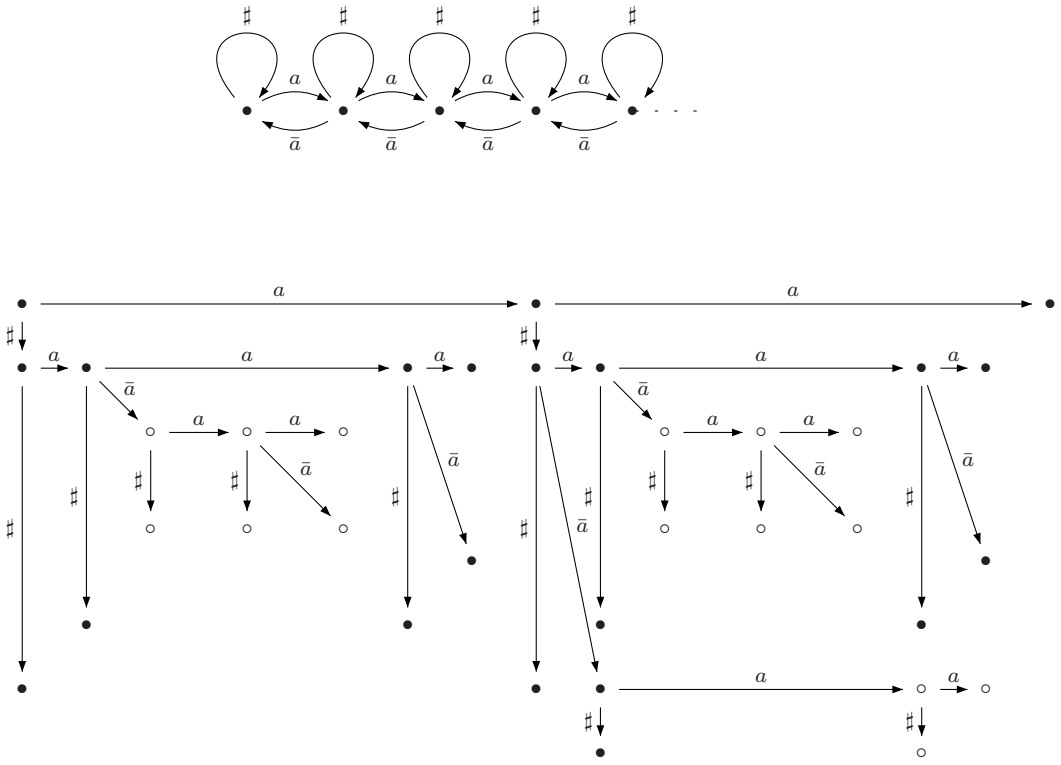


FIG. 3.4 – *Illustration de la construction du lemme 3.3.2sur la demi-droite. .*

affirme que l'opération de Treegraph est MSO-compatible. La première preuve complète de ce résultat est donnée par Walukiewicz dans [Wal96a, Wal02].

Théorème 3.3.4 ([CW98, Wal02])**.** *Le dépliage depuis un sommet MSO-définissable et le* Treegraph *sont des transformations MSO-compatibles.*

Remarque 3.3.5. Le fait que le sommet à partir duquel s'effectue le dépliage soit définissable en logique monadique est crucial. En effet, il est possible de définir une forêt dénombrables de théorie monadique décidable contenant un arbre de théorie monadique indécidable. La racine r_0 de cet arbre n'est bien entendu pas MSO-définissable. Si l'on déplie cette forêt à partir de r_0, nous obtenons un arbre de théorie monadique indécidable.

Nous allons maintenant donner la construction d'une telle forêt. Fixons un ensemble d'étiquettes Σ et un arbre t_{ind} de théorie monadique indécidable étiqueté par Σ. À une équivalence syntaxique près (voir par exemple [EF95]), il n'y a qu'un nombre fini de formules monadiques de rang de quantification au plus k. Nous noterons $F^{\Sigma}_{\leq k}$ un ensemble fini de représentants de ces classes d'équivalence. Pour tout graphe G sur Σ, nous notons $\text{Thm}^{\Sigma}_{\leq k}(G)$ l'ensemble des formules de $F^{\Sigma}_{\leq k}$

satisfaites par G.

Intuitivement, la forêt $\mathcal{F}$ contient pour tout $k \geq 1$ et pour tout sous-ensemble $M \subseteq \mathrm{Thm}^{\Sigma}_{\leq k}$ tel que M est réalisable par un arbre déterministe t_M sur Σ, une infinité dénombrable de copies disjointes de t_M. Formellement, la forêt $\mathcal{F}$ est l'union disjointe suivante:

$$t_{\mathrm{ind}} \uplus \biguplus_{k\geq 1} \biguplus_{M\subseteq \mathrm{Thm}^{\Sigma}_{\leq k}} \biguplus_{i\geq 1} t_M$$

où t_M est un arbre déterministe sur Σ dénombrable tel que $\mathrm{Thm}^{\Sigma}_{\leq k}(t_M) = M$ s'il existe et est l'ensemble vide sinon.

Fait 1. La forêt $\mathcal{F}$ a une théorie monadique décidable.

Par [She75], il résulte que pour décider de la théorie monadique de $\mathcal{F}$, il suffit de décider les formules $\varphi^{\geq \ell}_M$ avec $\ell \geq 1$ et $M \subseteq \mathrm{Thm}^{\Sigma}_k$ qui expriment que $\mathcal{F}$ contient au moins ℓ arbres distincts $t_1, \ldots, t_\ell$ tels que $\mathrm{Thm}^{\Sigma}_k(t_1) = \ldots = \mathrm{Thm}^{\Sigma}_k(t_n) = M$. Par construction de $\mathcal{F}$, si M est réalisable alors il existe une infinité dénombrable d'arbres t tels que $\mathrm{Thm}^{\Sigma}_k(t) = M$. Il suffit donc de savoir décider si un type M est réalisable par un arbre.
D'après le théorème de Rabin, pour tout k et pour tout $M \subset \mathrm{Thm}^{\Sigma}_{\leq k}$, nous pouvons décider s'il existe un arbre déterministe t étiqueté par Σ et tel que $\mathrm{Thm}^{\Sigma}_{\leq k}(t) = M$. Donc par construction de $\mathcal{F}$, pour décider si $\mathcal{F}$ satisfait une formule $\varphi^{\geq \ell}_M$, il suffit de décider si M est réalisable par un arbre déterministe sur Σ. Si tel est le cas alors $\mathcal{F}$ satisfait $\varphi^{\geq \ell}_M$ sinon $\mathcal{F}$ ne satisfait pas cette formule.

3.4 Résultats de commutation partielle

Dans ce paragraphe, nous présentons des résultats de commutation partielle entre les interprétations et les transductions monadiques d'un coté et le dépliage et le Treegraph de l'autre.

Un premier résultat simple de commutation, déjà présenté dans [Blu03], est valable entre une classe restreinte d'interprétations monadiques et le Treegraph.

Proposition 3.4.1. *Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$, pour toute interprétation $\mathcal{I}$ et pour tout symbole $\sharp \in \Lambda$ qui n'apparaît pas dans les formules définissant $\mathcal{I}$, il existe une interprétation monadique $\mathcal{J}$ telle que pour tout graphe connexe G sur (Σ,C) avec $\sharp \notin \Lambda_G$, nous avons*

$$\mathrm{Treegraph}(\mathcal{I}(G),\sharp) = \mathcal{J}(\mathrm{Treegraph}(G,\sharp)).$$

Démonstration. Soient Σ,Γ deux sous-ensembles finis de Λ et C,D deux sous-ensembles finis de Θ et $\mathcal{I} = ((\varphi_a(x,y))_{a\in\Gamma},(\varphi_c(x))_{c\in D})$ une interprétation monadique. Considérons l'interprétation monadique $\mathcal{J} = ((\psi_a(x,y))_{a\in\Gamma},(\psi_c(x))_{c\in D})$

où:

$$\begin{array}{ll} \psi_a(x,y) = \overline{\varphi_a}(x,y) \wedge \mathrm{Reach}_{\Sigma\cup\bar{\Sigma}}(x,y) & \text{for } a \in \Gamma \\ \psi_c(x) = \overline{\varphi_c}(x) & \text{for } c \in D \end{array}$$

où $\mathrm{Reach}_{\Sigma\cup\bar{\Sigma}}(x,y)$ est une formule telle que pour tout G et tous sommets $u,v \in V_G$, $G \models \mathrm{Reach}_{\Sigma\cup\bar{\Sigma}}[u,v]$ si et seulement si $u \xrightarrow[G]{w} v$ pour un certain $w \in (\Sigma \cup \bar{\Sigma})^*$ et où $\overline{\varphi_a}(x,y)$ (resp. $\overline{\varphi_c}$) est obtenue en restreignant les quantificateurs de $\varphi_a(x,y)$ (resp. $\varphi_c(x)$) aux ensembles de sommets qui ne contiennent que des accessibles à partir de x par un chemin étiqueté dans $(\Sigma \cup \bar{\Sigma})^*$.

Formellement, pour toute formule φ contenant des variables libres, nous définissons par récurrence une formule $\overline{\varphi}$ contenant le même ensemble de variables libres avec éventuellement l'ajout de la variable x comme suit:

$$\begin{array}{lcl} \overline{\varphi} & = & \varphi \quad \text{pour } \varphi \text{ atomique} \\ \overline{\varphi \vee \psi} & = & \overline{\varphi} \vee \overline{\psi} \\ \overline{\neg\varphi} & = & \neg\overline{\varphi} \\ \overline{\forall X\, \varphi} & = & \forall X\, (\forall z, z \in X \rightarrow \mathrm{Reach}_{\Sigma\cup\bar{\Sigma}}(z,x)) \rightarrow \overline{\varphi} \\ \overline{\forall z\, \varphi} & = & \forall z\, \mathrm{Reach}_{\Sigma\cup\bar{\Sigma}}(z,x) \rightarrow \overline{\varphi} \end{array}$$

Soit G un graphe sur (Σ,C) et $\sharp \in \Lambda - \Sigma$. Pour tout $u,v \in V_G^+$ et $a \in \Gamma$, $\mathrm{Treegraph}(G,\sharp) \models \psi_a[u,v]$ si et seulement si $u = wu'$ et $v = wv'$ pour $w \in V_G^*$ et $u',v' \in V_G$ et si $G \models \varphi_a[u',v']$. Il s'en suit alors que $\mathcal{J}(\mathrm{Treegraph}(G,\sharp)) = \mathrm{Treegraph}(\mathcal{I}(G),\sharp)$. □

La proposition suivante montre que le coloriage rationnel «commute» avec le dépliage.

Proposition 3.4.2. *Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$ et pour tout coloriage rationnel μ et pour tout graphe sur G (Σ,C) et pour tout sommet $r \in V_G$, il existe une transduction monadique $\mathcal{T}$ telle que:*

$$\mu_r(\mathrm{Unf}(G,r)) \approx \mathrm{Unf}(\mathcal{T}(G),r')$$

pour un certain $r' \in V_{\mathcal{T}(G)}$. De plus, si r est MSO-definissable dans G alors r' est effectivement[1] *MSO-definissable dans $\mathcal{T}(G)$. Si G est déterministe alors $\mathcal{T}(G)$ l'est aussi.*

Démonstration. Soit $\Sigma \subset \Lambda$ et $C \subset \Theta$ deux ensembles finis et soit μ un coloriage rationnel et $D = \mathrm{Dom}(\mu) = \{d_1 \ldots d_n\}$ pour $n \geq 1$.

Pour tout $i \in [1,n]$, notons R_i l'ensemble $\mu(d_i) \in \mathrm{Rat}((\Lambda \cup \Theta)^*)$ et notons $\Gamma \subset \Lambda$ (resp. $E \subset \Theta$) l'ensemble fini de symboles de Λ (resp. Θ) apparaissant dans $\bigcup_{i\in[1,n]} R_i$ ou dans Σ (resp. ou dans C).

1. La formule $\psi(x)$ définissant r' peut être effectivement calculée à partir de la formule $\varphi(x)$ définissant r.

Pour tout graphe T sur (Σ,C) qui soit un arbre de racine $r \in V_T$, nous associons à chaque sommet $v \in V_T$ un mot $\tau_u \in 2^E(\Gamma 2^E)^*$ en prenant $\tau_r = \Theta_T(r) \cap E$ et $\tau_u = \tau_v a(\Theta_T(u) \cap E)$ si $(v,a,u) \in T$.

Fait 1. Il existe un graphe fini A sur (Γ,E) et des ensembles I, $F_{d_1},\ldots,F_{d_n} \subseteq V_A$ tels que pour tout arbre t sur (Σ,C) et pour tout $u \in \mathrm{Dom}(t)$ et pour tout $\ell \in [1,n]$,

$$d_i \in \Theta_{\mu_r(T)}(u) \text{ ssi } i \xrightarrow[A]{\tau_u} q \text{ pour un certain } i \in I \text{ et } q \in F_{d_i}.$$

Pour tout $i \in [1,n]$, notons $\mathcal{L}_i$ l'ensemble (fini) des résidus à gauche de R_i et notons enfin $\mathcal{L} = 2^{\mathcal{L}_1} \times \ldots \times 2^{\mathcal{L}_n}$. Les sommets de A appartiennent à l'ensemble $\mathcal{V} = \mathcal{L} \times 2^E$ et le graphe fini A sur (Γ,E) est défini par :

$$\{((V_1,\ldots,V_n),F),a,((V'_1,\ldots,V'_n),F')) \in \mathcal{V} \times \Gamma \times \mathcal{V} \mid V'_i = \textstyle\bigcup_{w \in a(F')^*} w^{-1}V_i\}$$
$$\cup \{(c,(V,F)) \mid (V,F) \in \mathcal{V} \text{ et } c \in F\}$$

Pour tout ensemble $F \subseteq E$, nous notons i_F l'élément

$$((\bigcup_{w \in F^*} w^{-1}(R_1),\ldots,\bigcup_{w \in F^*} w^{-1}(R_n)),F)$$

de $\mathcal{V}$ et nous prenons $I = \{i_F \mid F \subseteq E\}$. Pour tout $i \in [1,n]$, nous définissons $F_i \subseteq \mathcal{V}$ comme l'ensemble de tous les $((V_1,\ldots,V_n),F) \in \mathcal{L} \times 2^E$ tels qu'il existe $L \in V_i$ avec $\varepsilon \in L$.

Le graphe A satisfait les propriétés suivantes:

- pour tout $F \subset E$, il existe un unique $u \in I$ avec $\Theta_A(u) = F$
- pour tout $a \in \Gamma$, $C \subseteq E$ et $u \in V_A$, il existe un unique $v \in V_A$ tel que $(u,a,v) \in A$ et $\Theta_A(v) = C$.

Fait 2. Pour tout graphe G sur (Σ,C), le graphe H dont l'ensemble des sommets est $V = \{(u,v) \mid \Theta_G(u) = \Theta_A(v)\} \subseteq V_G \times V_A$ et qui est défini par:

$$\begin{array}{rcl} H & = & \{((u_1,v_1),a,(u_2,v_2)) \in V \times \Sigma \times V \mid (u_1,a,u_2) \in G \text{ et } (v_1,a,v_2) \in A\} \\ & \cup & \{(d_i,(u,v)) \in D \times V \mid v \in F_{d_i}\} \end{array}$$

est définissable dans G par une transduction monadique $\mathcal{T}_0$. De plus, si G est déterministe, il suit des propriétés de A que H est aussi déterministe.

Fait 3. Pour tout graphe G sur (Σ,C) et $r \in V_G$, si nous notons i_0 l'unique élément de I tel que $\Theta_A(i_0) = \Theta_G(r)$ alors $\mathrm{Unf}(H,(r,i_0)) \approx \mu_r(\mathrm{Unf}(G,r))$.

Par construction de A et de H, nous avons pour tout sommet $\pi = u_0a_1\ldots a_nu_n$ de $V_{\mathrm{Unf}(G,r)}$, le sommet $\pi' = (u_0,v_0)a_1\ldots a_n(u_n,v_n)$ appartient à $V_{\mathrm{Unf}(H,(r,i_0))}$ si et seulement si pour tout $j \leq n$, $i_0 \xrightarrow[A]{\tau_{\pi_j}} v_j$ où $p_j = u_0a_1\ldots a_ju_j$.

Si r est MSO-définissable, nous modifions $\mathcal{T}$ pour qu'elle ajoute un arc entrant sur le sommet (r,i_0) étiqueté par un nouveau symbole de Λ et venant d'un nouveau sommet. Cet ajout ne change pas le dépliage de H depuis (r,i_0) et rend (r,i_0) MSO-définissable. □

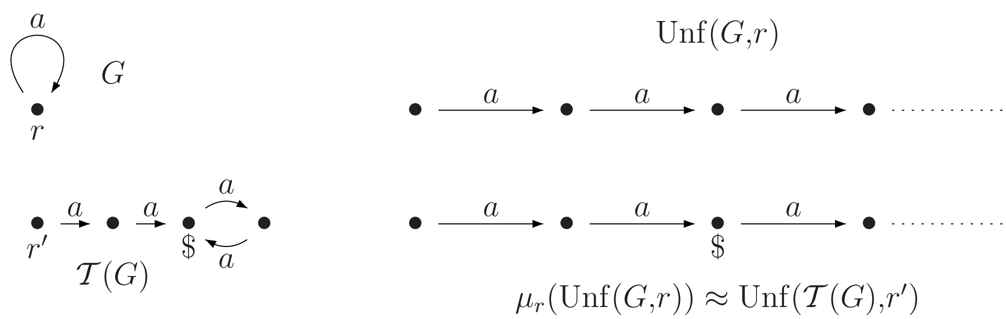


FIG. 3.5 – *Illustration de la proposition 3.4.2*

Remarque 3.4.3. Dans la proposition 3.4.2, la transduction monadique ne peut pas, dans le cas général, être remplacée par une interprétation monadique. Considérons, par exemple, le graphe G étiqueté par le singleton $\{ a \}$ présenté dans la figure 3.5 et considérons le coloriage rationnel μ défini par $\mu(\$) = aa(aa)^+$. Le graphe $\mathcal{T}(G)$ construit dans la proposition précédent est présenté. Il est évident qu'il n'existe par d'interprétation $\mathcal{I}$ tel que $\text{Unf}(\mathcal{I}(G),r) \approx \mu_\varepsilon(\text{Unf}(G,r))$. En effet, quelque soit l'interprétation $\mathcal{I}$, $\mathcal{I}(G)$ possède un unique sommet.

Proposition 3.4.4. *Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$ et tout coloriage monadique $\mathcal{M}$, il existe un coloriage monadique $\mathcal{M}'$ et un coloriage rationnel μ tels que pour tout graphe déterministe G sur (Σ,C) et pour tout sommet $r \in V_G$,*

$$\mathcal{M}(\text{Unf}(G,r)) = \mu_r(\text{Unf}(\mathcal{M}'(G),r)).$$

Démonstration. Soient $\Sigma \subset \Lambda$ et $C,D \subset \Theta$ des ensembles finis et soit $\mathcal{M} = (\varphi_d(x))_{d\in D}$ un coloriage monadique. Pour tout $d \in D$, notons $A_d = (Q_d \times 2^{\{0\}} \times 2^{\{0\}},I_d,\Delta_d,\Omega_d)$ un automate d'arbre à parité normalisé sur les arbres déterministes sur (Σ,C) acceptant $\varphi_d(x)$. Nous supposons sans perte de généralité que pour tout $d \neq d' \in D$, $\Delta_d \cap \Delta_{d'} = \emptyset$ et nous notons $\Delta = \bigcup_{d\in D} \Delta_d$. Considérons $\widetilde{\Delta} \subset \Theta$ un ensemble disjoint de Δ et de C et en bijection avec Δ. Pour tout $\delta \in \Delta$, nous noterons $\tilde{\delta}$ le symbole correspondant dans $\widetilde{\Delta}$.

Considérons un coloriage monadique $\mathcal{N}$ par $\widetilde{\Delta}$ des graphes sur (Σ,C) tel que pour tout arbre déterministe t sur (Σ,C) et pour tout $u \in \text{Dom}(t)$ et $\delta \in \Delta_d$, $(\tilde{\delta},u) \in \mathcal{N}(t)$ si et seulement si il existe une exécution acceptante de A_d sur $t_{/u}$ commençant par la transition δ.

La preuve procède en deux étapes. Tout d'abord, nous définissons un coloriage rationnel μ tel que pour tout arbre déterministe t sur (Σ,C), $\mathcal{M}(t) = \mu_\varepsilon(\mathcal{N}(t))$. Dans une seconde étape, nous construisons un coloriage monadique

des graphes sur (Σ,C) tel que pour tout (Σ,C)-graphe déterministe G et tout $r \in V_G$, $\mathcal{N}(\mathrm{Unf}(G,r)) = \mathrm{Unf}(\mathcal{M}'(G),r)$.

Première étape. Pour tout $d \in D$, nous définissons un ensemble rationnel $R_d \subseteq \widetilde{\Delta}(\Sigma\widetilde{\Delta})^*$ tel que pour tout arbre déterministe t sur (Σ,C) et tout $u \in \mathrm{Dom}(t)$, $\varepsilon \xrightarrow[\mathcal{N}(t)]{w} u$ pour un certain $w \in R_b$ si et seulement si $t \models \varphi_d[u]$.

Pour tout $\delta \in \Delta$, nous écrirons $\delta = (q_\delta,f_\delta)$ et $q_\delta = (p_\delta,M_\delta,N_\delta)$. Pour tout $d \in D$, nous prenons R_d comme l'ensemble des mots $\tilde{\delta_0}a_0 \dots a_n\tilde{\delta_n} \in \widetilde{\Delta_d}(\Sigma\widetilde{\Delta})^*$ avec $n \geq 0$ tel que pour tout $i \in [0,n-1]$, $M_{\delta_i} = \emptyset, N_{\delta_i} = \{1\}$, $f_{\delta_i}(a_i) = q_{\delta_{i+1}}$, $q_{\delta_0} \in I_d$ et $M_{\delta_n} = N_{\delta_n} = \{1\}$.

Pour tout arbre déterministe t sur (Σ,C), tout $d \in D$, et pour tout $u = a_1 \dots a_{|u|} \in \mathrm{Dom}(t)$ si $t \models \varphi_d[u]$ alors il existe une exécution ρ de A_d acceptant u. Prenons $w = \widetilde{\Phi_\rho(u_0)}a_1 \dots a_n\widetilde{\Phi_\rho(u_{|u|})}$ où $u_j = a_1 \dots a_j$ pour $j \in [0,|u|]$. Par construction, w appartient à R_d et $\varepsilon \xrightarrow[\mathcal{N}(t)]{w} u$.

Réciproquement pour tout $w \in R_d$, s'il existe $u \in \mathrm{Dom}(t)$ tel que $\varepsilon \xrightarrow[\mathcal{N}(t)]{w} u$ alors il existe une exécution de A_d acceptant u. Ceci est établi par récurrence sur la longueur de w.

Deuxième étape. Pour tout graphe déterministe G sur (Σ,C) et pour tout $\delta \in \Delta_d$, nous définissons une formule $\varphi_\delta(x)$ telle que pour tout $u \in V_G$, A_d ait une exécution acceptante sur $\mathrm{Unf}(G,u)$ commençant par δ.

Soient $\underline{\Delta}$ et $\underline{\underline{\Delta}} \subset \Lambda$ des ensembles disjoints de Σ et de Δ mais en bijection avec Δ. Nous supposons que $\underline{\Delta}$ et $\underline{\underline{\Delta}}$ sont disjoints et pour tout $\delta \in \Delta$, nous noterons $\underline{\delta}$ (resp. $\underline{\underline{\delta}}$) le symbole correspondant dans $\underline{\Delta}$ (resp. dans $\underline{\underline{\Delta}}$). Nous considérons le jeu de parité G_d défini par:

$$\begin{aligned}
G_d \;=\;& \{((q,u),\underline{\delta},(\delta,u)) \in V_0 \times \underline{\Delta} \times V_1 \mid \delta = (q,f)\} \\
\cup\;& \{((\delta,u),a,(q,ua)) \in V_1 \times \Sigma \times V_0 \mid \delta = (p,f), a \in \mathrm{Dom}(f) \text{ et } q = f(a)\} \\
\cup\;& \{(u,\underline{\underline{\delta}},(u,\delta)) \in V_G \times \underline{\underline{\Delta}} \times V_1 \mid (u,\delta) \in V_1\} \\
\cup\;& \{(p_i,v),(0,v) \mid v = (q,u) \in V_0 \cap V_{G_d} \text{ et } i = \Omega(q)\} \\
\cup\;& \{(p_i,v),(1,v) \mid v = ((q,f),u) \in V_1 \cap V_{G_d} \text{ et } i = \Omega(q)\} \\
\cup\;& \{(p_0,v),(0,v) \mid v \in V_G\}
\end{aligned}$$

où $V_0 = Q \times V_G$ et $V_1 = \{(\delta,u) \mid \delta = (q,f), \forall a\Sigma, \exists v \in V_G(u,a,v) \in G \Leftrightarrow a \in \mathrm{Dom}(f)\} \subseteq \Delta_d \times V_G$.

Il suit du lemme 1.4.6 que pour tout $\delta \in \Delta_d$, le joueur 0 gagne G_d depuis $(\delta,u) \in V_{G_d}$ si et seulement si A_d admet une exécution acceptante sur $\mathrm{Unf}(G,u)$ commençant par δ.

Le jeu G_d est définissable dans G par une transduction monadique $\mathcal{T}$. Plus précisément, $\mathcal{T}(G)$ est isomorphe à G_d et il existe un morphisme h de $\mathcal{T}(G)$ à G_d tel que pour tout sommet $v \in V_G \in V_{\mathcal{T}(G)}$, $h(v) = v$. Par la proposition 1.4.7, il existe une formule $\varphi_0(x)$ telle que pour tout $u \in V_G$, $G_d \models \varphi_0[u]$ si et seulement

si le joueur 0 gagne G_d à partir de u. Considérons la formule $\psi_\delta(x) = \exists y, E_{\underline{\delta}} xy \wedge \varphi_0(y)$. Notons que pour tout $u \in V_{\mathcal{T}(G)}$, $\mathcal{T}(G) \models \psi_\delta[u]$ si et seulement si $u \in V_G$ et s'il existe une exécution acceptante de A_d sur $\mathrm{Unf}(G,u)$ commençant par δ.

Par la proposition 3.1.3, il existe une formule $\psi_\delta(x)$ telle que pour tout $u \in V_G$, $G \models \psi_\delta[u]$ si et seulement si $\mathcal{T}(G) \models \varphi_\delta[u]$.

Enfin, nous définissons le coloriage monadique $\mathcal{M}'$ sur les graphes (Σ,C) par $(\psi_\delta(x))_{\delta \in \Delta_d}$. □

En combinant les propositions 3.4.4 et 3.4.2, nous obtenons le résultat suivant.

Corollaire 3.4.5. *Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$, et tout coloriage monadique $\mathcal{M}$ des graphes sur (Σ,C), il existe une transduction monadique $\mathcal{T}$ telle que pour tout graphe déterministe G et pour tout sommet $r \in V_G$,*

$$\mathcal{M}(\mathrm{Unf}(G,r)) = \mathrm{Unf}(\mathcal{T}(G),r')$$

où $r' \in V_{\mathcal{T}(G)}$. De plus, si r est définissable dans G alors r' est effectivement définissable dans $\mathcal{T}(G)$.

En combinant les propositions 3.2.1, 3.4.4 et 3.4.2, nous obtenons le corollaire ci-dessous.

Corollaire 3.4.6. *Pour tous ensembles finis $\Sigma \subset \Lambda$ et $C \subset \Theta$ et toute interprétation monadique $\mathcal{I}$, il existe une substitution rationnelle colorée h et une transduction monadique $\mathcal{T}$ telle que pour tout graphe déterministe G et pour tout $r \in V_G$,*

$$\mathcal{I}(\mathrm{Unf}(G,r)) = h^{-1}(\mathrm{Unf}(\mathcal{T}(G),r'))$$

où $r' \in V_{\mathcal{T}(G)}$. De plus, si r est définissable dans G alors r' est effectivement définissable dans $\mathcal{T}(G)$.

Remarquons que comme la copie par un graphe fini commute avec le dépliage, l'interprétation $\mathcal{I}$ peut être remplacée par une transduction dans le corollaire précédent.

3.5 Préservation de la propriété de sélection

Dans ce paragraphe, nous présentons quelques cas de transfert de la propriété de sélection. Des résultats similaires ont été obtenus dans [Fra05].

L'interprétation monadique ne préserve pas la propriété de sélection car elle peut par exemple introduire du non déterminisme. Cependant la proposition suivante donne des conditions suffisantes pour que l'application inverse de l'interprétation monadique transfère la propriété de sélection.

Proposition 3.5.1. *Pour tous graphes G et H avec $V_H \subseteq V_G$ tels que G possède la propriété de sélection et tels qu'il existe deux interprétations monadiques $\mathcal{I}$ et $\mathcal{J}$ avec $H = \mathcal{I}(G)$ et $G|_{V_H} = \mathcal{J}(H)$ alors H possède la propriété de sélection.*

Démonstration. Soient G et H deux graphes avec $V_H \subseteq V_G$. Supposons que G possède la propriété de sélection et qu'il existe deux interprétations monadiques $\mathcal{I}$ et $\mathcal{J}$ telles que $H = \mathcal{I}(G)$ et $G|_{V_H} = \mathcal{J}(H)$. Cette dernière égalité implique que V_H est effectivement définissable dans G. Nous pouvons donc construire une formule $\varphi_0(X)$ telle que $G \models \exists^{=1} X, \varphi_0(X)$ et tel que $G \models \varphi_0[V_H]$.
Montrons que H possède la propriété de sélection.

Soit $\varphi(X)$ une formule monadique. Par la proposition 3.1.3, il existe une formule monadique $\varphi_{\mathcal{J}}(X)$ telle que pour tout $U \subset V_H$, $H \models \varphi[U]$ si et seulement si $G \models \varphi_{\mathcal{J}}[U]$. Considérons la formule $\varphi'(X) = \exists Y, \varphi_0(Y) \wedge X \subseteq Y \wedge \varphi_{\mathcal{J}}(X)$. Nous avons:

$$H \models \exists X, \varphi(X) \;\Leftrightarrow\; G \models \exists X, \varphi'(X).$$

Comme G satisfait la propriété de sélection, nous pouvons construire un sélecteur $\psi(X)$ pour la formule $\varphi'(X)$. Par la proposition 3.1.3, nous pouvons construire une formule $\psi_{\mathcal{J}}(X)$ telle que pour tout $U \subseteq V_H$,

$$G \models \psi[U] \;\Leftrightarrow\; H \models \psi_{\mathcal{J}}[U].$$

Il est aisé de vérifier que $\psi_{\mathcal{J}}(X)$ est un sélecteur de $\varphi(X)$. Le graphe H satisfait donc la propriété de sélection. □

Une conséquence de cette proposition est que si deux graphes G et H ont le même ensemble de sommets et si G est interprétable dans H et si H est interprétable dans G alors G satisfait la propriété de sélection si et seulement si H la satisfait.

Proposition 3.5.2. *Pour tout graphes G et H avec $V_G = V_H$ tel qu'il existe deux interprétations monadiques $\mathcal{I}$ et $\mathcal{J}$ avec $H = \mathcal{I}(G)$ et $G = \mathcal{J}(H)$ alors G a la propriété de sélection si et seulement si H a la proposition de sélection.*

Dans le cas des arbres déterministes, le Treegraph transfère la propriété de sélection. Cette propriété repose en grande partie sur le lemme 3.3.2.

Proposition 3.5.3. *Pour tout arbre déterministe T possédant la propriété de sélection et tout symbole $\sharp \in \Lambda \setminus \Lambda_T$, le graphe* Treegraph($T$,$\sharp$) *possède la propriété de sélection.*

Démonstration. Soit T un arbre déterministe sur (Σ,C) de racine r possédant la propriété de sélection et soit $\sharp$ un symbole $\Lambda \setminus \Lambda_T$. Pour simplifier la présentation, nous traitons le cas où T n'est pas coloré (*i.e.* $C = \emptyset$). Le cas coloré est similaire.

Reprenons la construction du lemme 3.3.2. Il existe deux substitutions rationnelles h_1 et h_2 et une restriction monadique $\mathcal{R}$ telle que:

$$\text{Treegraph}(T,\sharp) \approx h_2^{-1}(\mathcal{R}(\text{Unf}(h_1^{-1}(T),r))).$$

Nous notons $\Sigma^\sharp$ l'ensemble $\Sigma \cup \overline{\Sigma} \cup \{\,\sharp\,\}$. Rappelons que $\mathcal{R}$ restreint le dépliage $\text{Unf}(h_1^{-1}(T),r)))$ aux sommets x dont l'étiquette de l'unique chemin allant de la racine à x ne contient pas de facteurs de la forme $x\bar{x}$ ou $\bar{x}x$ quelque soit $x \in \Sigma$. Notons $\overline{T}$ le graphe $h^{-1}(T)$ et T_0 l'arbre $\mathcal{R}(\text{Unf}(h_1^{-1}(T),r))$. Ces deux graphes sont étiquetés par $\Sigma^\sharp$. À tout $u \in V_{T_0}$, nous associons un symbole, noté Σ_u, dans $\Sigma^\sharp \cup \{\,\varepsilon\,\}$ défini par:

$$\Sigma_u = \begin{cases} \varepsilon & \text{si } u \text{ est la racine } T, \\ a & \text{s'il existe un nœud } v \in V_{T_0} \text{ tel que } (v,a,u) \in T. \end{cases}$$

De plus, pour tout $u \in V_{T_0} \subseteq V_T^+$, nous noterons $\text{Rep}(u)$ le dernier sommet V_T dans la suite u. Intuitivement, $\text{Rep}(u)$ est le sommet de T dont est issu u dans le dépliage de $\overline{T}$.
Enfin, nous notons G le graphe $h_2^{-1}(T_0)$ qui est isomorphe à $\text{Treegraph}(T,\sharp)$.

Avant de passer au cœur de la démonstration, établissons quelques propriétés liant ces différents graphes.

Fait 1. Les graphes T_0 et G ont le même ensemble de sommets et il existe deux interprétations $\mathcal{I}$ et $\mathcal{J}$ telles que $G = \mathcal{I}(T_0)$ et $T_0 = \mathcal{J}(G)$.

Par définition de h_2^{-1}, $V_{T_0} = V_G$. Comme nous l'avons déjà mentionné, les substitutions rationnelles sont des cas particuliers d'interprétations monadiques. L'existence de $\mathcal{I}$ est donc immédiate. Considérons l'interprétation monadique $\mathcal{J} = ((\varphi_a(x,y))_{a\in\Sigma},(\varphi_c(x))_{c\in C})$ où:

$$\begin{array}{lcll} \varphi_a(x,y) & = & E_a(x,y) \wedge \neg(\exists z,z',E_\sharp(z',z) \wedge \text{Reach}_\Sigma(x,z)) & \text{pour } a \in \Sigma, \\ \varphi_{\bar{a}}(x,y) & = & E_a(y,x) \wedge \exists z,z',E_\sharp(z',z) \wedge \text{Reach}_\Sigma(x,z) & \text{pour } a \in \Sigma, \end{array}$$

où $\text{Reach}_\Sigma(x,y)$ exprime l'existence d'un chemin de x à y étiqueté dans Σ^*. Il est aisé de vérifier $\mathcal{J}(G) = T_0$.

Fait 2. Le graphe $\text{Treegraph}(T,\sharp)$ possède la propriété de sélection si et seulement si T_0 la possède.

Par la proposition 3.5.1 et par le fait 1, T_0 possède la propriété de sélection si et seulement si G la possède. Comme G est isomorphe à $\text{Treegraph}(T,\sharp)$, la propriété annoncée est établie.

Montrons maintenant que T_0 satisfait la propriété de sélection. Considérons pour cela une formule monadique $\varphi(X)$. Il existe un automate d'arbres à parité $A = (Q,\{\,i_0\,\},F,\Delta,\Omega)$ et $Q_X \subseteq Q$ tels que:

- s'il existe une exécution acceptante ρ de A sur T_0 alors $T_0 \models \varphi[U]$ où U est l'ensemble des sommets u de T_0 tels que $\rho(u) \in Q_X$.

– réciproquement, s'il existe $U \subseteq V_{T_0}$ tel que $T_0 \models \varphi[U]$ alors il existe une exécution acceptante de A sur T_0 telle que U soit l'ensemble des sommets $u \in V_{T_0}$ tels que $\rho(u) \in Q_X$.

L'ensemble des transitions Δ peut s'écrire $\{\delta_1,\ldots,\delta_n\}$ pour un certain $n \geq 0$. Nous supposerons que $\Omega(Q) \subseteq [1,N]$. Nous noterons pour tout $i \in [1,n]$, δ_i est égale à (p_i,f_i) où $p_i \in Q$ et f_i est une fonction partielle de $\Sigma^\sharp \dashrightarrow Q$.

Intuitivement pour construire un sélecteur $\psi(X)$ de $\varphi(X)$ sur T_0, nous allons sélectionner une stratégie gagnante de l'automate A sur T_0.

Fait 3. Si A accepte T_0 alors il existe une exécution acceptante ρ de A sur T_0 telle que pour tous $u,v \in V_{T_0}$, $\rho(u) = \rho(v)$, $\Sigma_u = \Sigma_v$ et $\mathrm{Rep}(u) = \mathrm{Rep}(v)$ alors $\Phi_\rho(u) = \Phi_\rho(v)$.

Considérons le jeu de parité G_A étiqueté par l'ensemble $\Sigma^\sharp \cup \Delta$ et coloré par $\{0,1,p_0,\ldots,p_N\}$ et défini par:

$$\begin{array}{rcl} G_A & = & \{((q,a,u),\delta,(\delta,a,u)) \in V_0 \times \Delta \times V_1 \mid \delta = (q,f)\} \\ & \cup & \{((\delta,b,u),a,(f(a),a,v)) \in V_1 \times \Sigma^\sharp \times V_0 \mid \delta = (p,f) \text{ et } u \xrightarrow[\overline{T}]{a} v\} \\ & \cup & \{(p_i,v),(0,v) \mid v = (q,a,u) \in V_0 \cap V_{G_A} \text{ et } i = \Omega(q)\} \\ & \cup & \{(p_i,v),(1,v) \mid v = (\delta,a,u) \in V_1 \cap V_{G_A}, f = (q,f) \text{ et } i = \Omega(q)\} \end{array}$$

où $V_0 = Q \times \Sigma^\sharp \times V_{T_0}$ et V_1 est l'ensemble des triplets (δ,a,u) avec $\delta = (q,f) \in \Delta$, $a \in \Sigma^\sharp \cup \{\varepsilon\}$ et avec:

$$\mathrm{Dom}(f) = \begin{cases} \{b \in \Sigma^\sharp \mid \exists v \in V_T, u \xrightarrow[\overline{T}]{b} v\} & \text{si } a \in \{\varepsilon,\sharp\} \\ \{b \in \Sigma^\sharp \mid b \neq \bar{a} \text{ et } \exists v \in V_T, u \xrightarrow[\overline{T}]{b} v\} & \text{si } a \in \Sigma \cup \overline{\Sigma}. \end{cases}$$

Il est aisé de vérifier que le joueur 0 gagne le jeu G_A depuis le sommet (i_0,ε,r) si et seulement si A accepte T_0. En effet, le dépliage de G_A depuis ce sommet est isomorphe à la composante connexe enraciné en (i_0,r) du jeu canonique $G_A^{T_0}$ associé à l'automate A sur le graphe T_0. Une stratégie positionnelle gagnante pour le joueur 0 sur G_A induit une exécution de A sur T_0 satisfaisant les propriétés annoncées.

Une exécution ρ satisfaisant les conditions du fait 3 peut être entièrement décrite sur T par une famille $\mathcal{S} = (U_i^a)_{\{a \in \Sigma^\sharp \cup \{\varepsilon\} \text{ et } i \in [1,n]\}}$ de sous-ensembles de V_T telle que pour tout $a \in \Sigma^\sharp \cup \{\varepsilon\}$ et pour tous $i,j \in [1,n]$, si $U_i^a \cap U_j^a \neq \emptyset$ alors $p_i \neq p_j$. Plus précisément, à de telle famille $\mathcal{S}$ de sous-ensembles est associée au plus une exécution $\rho_\mathcal{S}$ de A sur T_0 vérifiant que pour tout $u \in V_{T_0}$, $\Phi_{\rho_\mathcal{S}}(u) = \delta_i$ implique $\mathrm{Rep}(u) \in U_i^{\Sigma_u}$.

Fait 4. Nous pouvons construire une formule monadique $\varphi_0(\overline{X})$ où $\overline{X} = \{X_i^a \mid i \in [1,n] \text{ et } a \in \Sigma^\sharp \cup \{\varepsilon\}\}$ telle que pour toute famille $\mathcal{S} = (U_i^a)_{i\in[1,n] \text{ et } a\in\Sigma^\sharp\cup\{\varepsilon\}}$ de sous-ensembles de V_T, $T \models \varphi_0[\mathcal{S}]$ si et seulement si $\mathcal{S}$ induit une exécution acceptante $\rho_\mathcal{S}$ de A sur T_0.

Considérons un enrichissement du jeu G_A, noté $\overline{G_A}$, dont les arcs sont étiquetés par $Q \times (\Sigma^\sharp \cup \{\varepsilon\})$ et défini:

$$\overline{G_A} = G_A \cup \{(u,(q,a),(q,a,u)) \mid u \in V_T, q \in Q \text{ et } a \in \Sigma^\sharp \cup \{\varepsilon\}\}.$$

Il existe une transduction monadique $\mathcal{T}$ telle que $\mathcal{T}(T) \approx \overline{G_A}$. En adaptant la proposition 1.4.8, nous construisons $\varphi_0(\overline{X})$.

Comme T possède la propriété de sélection, nous pouvons construire un sélecteur de $\psi_0(\overline{X})$ de $\varphi_0(\overline{X})$.

Fait 5. Nous pouvons construire une formule $\varphi_1(X_1,\ldots,X_n)$ telle que $T_0 \models \exists^{\leq 1}X_1,\ldots,\exists^{\leq 1}X_n,\varphi_1(X_1,\ldots,X_n)$ et pour tous ensembles $U_1,\ldots,U_n \subseteq V_{T_0}$, $T_0 \models \varphi_1[U_1,\ldots,U_n]$ alors

- pour tout $i \neq j$, $U_i \cap U_j = \emptyset$,
- il existe une exécution acceptante ρ de A sur T_0 tel que pour tout $i \in [1,n]$, $\Phi_\rho(u) = \delta_i$ si et seulement si $u \in U_i$.

Commençons par construire une formule $\varphi_2(\overline{X})$ telle que Treegraph$(G,\sharp) \models \exists^{\geq 1}\overline{X}, \varphi_2(\overline{X})$ et telle que:

- Treegraph$(G,\sharp) \models \exists\overline{X}, \varphi_2(\overline{X})$ si et seulement si $G \models \exists\overline{X}, \psi_0(\overline{X})$,
- si pour une famille $\mathcal{S} = (U_i^a)_{i\in[1,n] \text{ et } a\in\Sigma^\sharp\cup\{\varepsilon\}}$, $T \models \psi_0[\mathcal{S}]$ alors nous avons Treegraph$(G,\sharp) \models \varphi_2[\mathcal{S}']$ où $\mathcal{S}'$ est la famille $(V_i^a)_{i\in[1,n] \text{ et } a\in\Sigma^\sharp\cup\{\varepsilon\}}$ où pour tout $i \in [1,n]$ et $a \in \Sigma^\sharp \cup \{\varepsilon\}$, V_i^a est l'ensemble de tous les sommets de Treegraph$(G,\sharp)$ (qui sont des suites de sommets de T) dont le dernier élément appartient à U_i^a.

Cette formule est construite par une adaptation de la méthode présentée dans la proposition 3.4.1. Comme T_0 est interprétable (à isomorphisme près) dans Treegraph$(G,\sharp)$ (cf. fait 2), nous pouvons construire une formule $\varphi_3(\overline{X})$ telle que:

- $T_0 \models \exists^{\leq 1}\overline{X}, \varphi_3(\overline{X})$
- $T_0 \models \exists\overline{X}, \varphi_3(\overline{X})$ si et seulement si $G \models \exists\overline{X}, \psi_0(\overline{X})$,
- si pour une famille $\mathcal{S} = (U_i^a)_{i\in[1,n] \text{ et } a\in\Sigma^\sharp\cup\{\varepsilon\}}$, $T \models \psi_0[\mathcal{S}]$ alors $T_0 \models \varphi_2[\mathcal{S}']$ où $\mathcal{S}'$ est la famille $(V_i^a)_{i\in[1,n] \text{ et } a\in\Sigma^\sharp\cup\{\varepsilon\}}$ où pour tout $i \in [1,n]$ et $a \in \Sigma^\sharp \cup \{\varepsilon\}$, V_i^a est l'ensemble de tous les sommets u de T_0 tels que Rep(u) appartient à U_i^a.

La formule $\varphi_1(\overline{X})$ est aisément construite à partir de $\varphi_3(\overline{X})$.

Nous pouvons maintenant construire un sélecteur pour la formule $\varphi(X)$. Il suffit de considérer la formule $\psi(X)$ définie par:

$$\exists X_1,\ldots,X_n,\, \varphi_1(X_1,\ldots,X_n) \wedge X = \bigcup_{\{i\in[1,n] \mid p_i\in Q_X\}} X_i.$$

Par le fait 5, $T_0 \models \exists^{\leq 1} X, \psi(X)$. De plus par ce même fait 5, nous avons que $T \models \exists X,\psi(X)$ si et seulement si A accepte T_0. Et donc par définition de A, $T_0 \models \exists X,\psi(X) \leftrightarrow \exists X,\varphi(X)$. Enfin supposons qu'il existe un ensemble $U \in V_{T_0}$ tel que $T_0 \models \psi[U]$. Par définition de $\psi(X)$, il existe donc $U_1,\ldots,U_n \subseteq V_{T_0}$ tels que $T_0 \models \varphi_1[U_1,\ldots,U_n]$. Par le fait 5, il existe une exécution acceptante ρ de A sur T_0 telle que pour tout $u \in V_{T_0}$, $\Phi_\rho(u) = \delta_i$ si et seulement si $u \in U_i$. Donc par définition de ψ, U est l'ensemble $\{u \in V_{T_0} \mid \rho(u) \in Q_X\}$. Et donc par définition de A, $T_0 \models \varphi[U]$. Nous avons établi que $T_0 \models \exists X,\psi(X) \rightarrow \varphi(X)$. La formule $\psi(X)$ est donc un sélecteur de la formule $\varphi(X)$. □

Nous ne savons pas si le Treegraph préserve la propriété de sélection pour un graphe quelconque. Cette question et d'autres liées à la propriété de sélection sera abordée nous l'espérons en collaboration avec Alexander Rabinovich.

Chapitre 4

Ensembles rationnels de piles de piles

Dans ce chapitre, nous introduisons la notion d'automates à pile de piles en tant qu'accepteurs de langages. Cette notion est apparue dans les années 70 et étend les automates à pile. Au niveau 2, les piles sont remplacées par des séquences de piles : des piles de piles. On peut comme au niveau 1 empiler ou dépiler le dernier symbole de la dernière pile de niveau 1. La nouvelle opération est la copie de la dernière pile de niveau 1 ainsi que sa destruction. Au niveau 3, les automates travaillent sur des piles de piles de piles qui sont des séquences de piles de piles. Les opérations de niveau au plus deux sont appliquées sur la dernière pile de niveau 2 et les nouvelles opérations sont la copie et la destruction de la dernière pile de niveau 2. Les automates à pile de piles sont ainsi définis pour tout niveau $k \geq 1$. Ils ont été introduits au niveau 2 comme des accepteurs pour les langages indexés, qui contiennent strictement les langages algébriques. Nous introduisons formellement les automates à pile de piles dans le paragraphe 4.1 où une présentation détaillée de l'historique de cette notion est donnée.

Pour les automates à pile, un résultat fondamental est que l'ensemble des contenus de pile accessibles, vus comme des mots sur l'alphabet de pile, forme un ensemble rationnel. Le but de ce chapitre est d'obtenir une description finie des ensembles de piles de piles accessibles par un automates à pile à partir de la configuration initiale et d'en étudier les propriétés algébriques et logiques.

Pour pouvoir obtenir une notion pertinente d'ensemble rationnel de piles de piles, il faut considérer un jeu d'opérations où la destruction inconditionnelle des piles est remplacée par une version plus symétrique où la dernière pile de niveau ℓ ne peut être détruite que si elle est égale à la précédente. Nous établissons dans le paragraphe 4.1 l'équivalence (en temps qu'accepteurs de langages) les automates à pile définis sur ces deux jeux d'opérations (cf. théorème 4.1.23). Nous montrons, dans le paragraphe 4.6, le manque de pertinence de la notion de rationalité induite

par le jeu d'opérations classiques et par là même, nous établissons la nécessité de considérer le jeu d'opérations symétriques.

Nous choisissons de qualifier d'ensemble rationnel de piles de niveau k tout ensemble qui peut être construit par application d'une suite rationnelle d'opérations symétriques de niveau k à la pile vide de niveau k. Il suit de la définition que les ensembles de piles de piles accessibles par un automate à pile de piles en partant de sa configuration initiale est régulier à notre sens. Cette notion est formellement introduite dans le paragraphe 4.2 où une fois les diverses notations nécessaires introduites, nous donnerons un plan détaillé du reste du chapitre.

Les paragraphes 4.3, 4.4 et 4.5 présentent différents accepteurs finis pour les ensembles rationnels de piles de niveau k. Nous étudions les propriétés algébriques et algorithmiques de cette notion et montrons que ces ensembles forment une algèbre de Boole (théorème 4.4.8). Nous définissons deux notions naturelles et complémentaires d'accepteurs déterministes pour ces langages ainsi que des procédures de déterminisation de complexité minimale (cf. théorème 4.4.15 et 4.5.21). De plus, dans le paragraphe 4.5, nous étendons à tout niveau la notion de relation préfixe-reconnaissable introduite dans [Cau96] et nous montrons que pour tout niveau k, les relations préfixe-reconnaissables de niveau k forment une algèbre de Boole (théorème 4.5.16).

Le paragraphe 4.7 montre l'équivalence entre les ensembles rationnels de piles de niveaux k et les ensembles définissables en logique monadique sur la structure canonique $\mathrm{GStacks}_k$ associée à ces piles (cf. théorème 4.7.4). De même, nous établissons que les relations définissables dans $\mathrm{GStacks}_k$ sont les relations préfixe-reconnaissables de niveau k (cf. théorème 4.7.5).

Enfin le paragraphe 4.8 applique les résultats sur les accepteurs finis des ensembles rationnels de niveau k à l'étude d'extensions des automates sur les mots telles que les automates bidirectionnels et les automates à galets.

La notion de rationalité a été étudiée et définie indépendamment par Fratani dans [Fra05]. Son approche est basée principalement sur la définissabilité en logique du second ordre monadique (cf. paragraphe 4.7). Une notion plus faible de rationalité a été introduite dans [BM04] qui forme aussi un algèbre de Boole mais qui n'est pas assez riche pour capturer les ensembles de configurations engendrés par un automate à pile. Nous verrons que cette notion apparaît naturellement dans notre étude (cf. paragraphe 4.6).

Une version préliminaire de ces résultats à été présentée dans [Car05].

4.1 Automates à pile de piles

Dans ce paragraphe, nous présentons les automates à pile de piles et les notions associées.

Dans les sous-paragraphes 4.1.1 et 4.1.2, nous définissons les piles de piles et les opérations les manipulant. Ces définitions sont classiques et suivent [KNU02]. Cependant, une différence notable est l'utilisation d'une version «symétrique» de la destruction des piles de niveau k notée $\overline{\text{copy}}_k$ au lieu de la destruction classique que nous noterons destr_k et qui est notée pop_{k+1} dans [KNU02].
Dans le sous-paragraphe 4.1.3, nous associons à chaque opération un symbole appelé instruction. Ceci nous permet de travailler de manière symbolique (*i.e.* dans le monoïde libre engendré par les instructions) sur les éléments du monoïde des opérations.
Dans le sous-paragraphe 4.1.4, nous caractérisons la plus petite suite d'opérations symétriques permettant de construire une pile de niveau k à partir de la pile vide de niveau k. Cette suite caractéristique jouera un rôle fondamental dans tous les résultats de ce chapitre.
Enfin, dans le sous-paragraphe 4.1.5, nous définissons les automates à pile de piles et montrons l'équivalence, du point de vue des langages acceptés, entre les automates à pile de piles définis avec le jeu d'opérations classiques et ceux définis avec le jeu d'opérations symétriques.

4.1.1 Pile de piles

Une *pile* de niveau 1 sur un alphabet fini Γ est représentée par un mot de Γ^*. La pile vide correspond au mot vide ε. L'ensemble de toutes les piles de niveau 1 sur l'alphabet Γ sera noté $\text{Stacks}_1(\Gamma) = \Gamma^*$. Pour clarifier notre propos, nous écrirons $[\,ABC\,]_1$ la pile correspondant au mot ABC et la pile vide sera notée $[\,]_1$.
Nous prenons pour convention que la dernière lettre du mot représente le haut de la pile. Nous définissons une fonction partielle top de $\text{Stacks}_1(\Gamma)$ dans Γ associant, à toute pile non vide, son dernier symbole (*i.e.* pour toute pile $[\,w\,]_1$, $\text{top}(w) = w(|w|)$ si $w \neq [\,]_1$ et est non définie sinon). Ainsi, le dernier symbole (ou symbole de haut de pile) de $[\,ABC\,]_1$ est $\text{top}([\,ABC\,]_1) = C$.

Pour tout $k > 1$, une *pile de niveau* k est une suite *non vide* de piles de niveau $k-1$. La pile vide de niveau k, notée $[\,]_k = \big[\,[\,]_{k-1}\,\big]_k$, est la pile de niveau k contenant uniquement la pile vide de niveau $k-1$. Nous noterons $[\,s_1,\ldots,s_n\,]_k$ la pile de niveau k correspondant à la suite $s_1,\ldots,s_n$ de piles de niveau $k-1$.
L'ensemble des piles de niveau k sur l'alphabet Γ est notée $\text{Stacks}_k(\Gamma)$ et est égal à $(\text{Stacks}_{k-1}(\Gamma))^+$. De même, nous noterons $\text{Stacks}_{\geq k}(\Gamma)$ l'ensemble des piles sur Γ de niveau au moins k. Enfin, l'ensemble des piles de tout niveau sur l'alphabet Γ sera noté $\text{Stacks}(\Gamma) = \bigcup_{k\geq 1} \text{Stacks}_k(\Gamma)$. Quand l'alphabet Γ se déduit du contexte, nous écrirons simplement Stacks_k, $\text{Stacks}_{\geq k}$ et Stacks au lieu de $\text{Stacks}_k(\Gamma)$, $\text{Stacks}_{\geq k}(\Gamma)$ et $\text{Stacks}(\Gamma)$.

Pour $k > 1$, l'application top_k de $\text{Stacks}_{>k}(\Gamma)$ dans Stacks_k associe à toute

pile de niveau plus grand que k sa dernière pile de niveau k. Elle est définie par:

$$\begin{cases} \mathrm{top}_k([s_1,\ldots,s_n]_{k+1}) &= s_n \\ \mathrm{top}_k([s_1,\ldots,s_n]_\ell) &= \mathrm{top}_k(s_n) \qquad \text{pour } \ell > k+1 \end{cases}$$

De manière analogue, nous étendons top en une fonction partielle de Stacks(Γ) dans Γ.

La pile de niveau 2 $[\,[\,ABC\,]\,[\,ABC\,]\,[\,CAB\,]\,]_2$ a pour plus haute pile de niveau 1 la pile $[\,CAB\,]_1$ et pour plus haut symbole $\mathrm{top}([\,[\,ABC\,]\,[\,ABC\,]\,[\,CAB\,]\,]_2) = \mathrm{top}([\,CAB\,]_1) = B$.

4.1.2 Opérations sur les piles de piles

Dans la suite de ce sous-paragraphe, nous fixons un alphabet de pile Γ. Une opération θ sur les piles de piles est une fonction partielle de Stacks(Γ) dans Stacks(Γ) qui préserve les niveaux des piles (*i.e.* pour tout $k \geq 1$, l'image d'une pile de niveau k est une pile de niveau k). Le niveau de l'opération θ, noté $|\theta|$ (s'il est défini), est le plus petit k tel que $\mathrm{Dom}(\theta) \cap \mathrm{Stacks}_k(\Gamma) \neq \emptyset$. La seule opération dont le niveau n'est pas défini est la fonction vide $\emptyset$ et, par convention, nous prendrons $|\emptyset| = +\infty$. Ainsi, pour toutes opérations θ et θ' nous avons $|\theta \cdot \theta'| \geq \max\{|\theta|,|\theta'|\}$.

Toutes les opérations que nous allons considérer respectent la politique d'accès aux piles de piles : dans une pile de niveau $k+1$, seule la plus haute pile de niveau ℓ peut être modifiée et ce quelque soit $\ell \in [1,k]$. Ainsi, une opération de niveau k quand elle est appliquée à une pile $s = \left[\, s_1,\ldots,s_{|s|} \,\right]_\ell$ de niveau $\ell > k$ s'applique sur la plus haute pile de niveau k de s (*i.e.* $\theta(s) = \left[\, s_1,\ldots,s_{|s|-1},\theta(s_{|s|}) \,\right]_\ell$).

Ainsi, pour définir une opération θ de niveau k, il est seulement nécessaire de la définir sur les piles de Stacks$_k$. Sa définition pour les niveaux $\ell > k$ est donnée récursivement par l'équation:

$$\theta([\,s_1,\ldots,s_n\,]_\ell) = [\,s_1,\ldots,\theta(s_n)\,]_\ell\,.$$

Opérations de niveau 1. Les opérations de niveau 1 sont les opérations push_x et pop_x permettant respectivement d'empiler ou de dépiler un symbole $x \in \Gamma$ sur la plus haute pile de niveau 1 et qui sont définies par:

$$\begin{array}{lcl} \mathrm{push}_x([\,s\,]_1) &=& [\,sx\,]_1 \\ \mathrm{pop}_x([\,s\,]_1) &=& \begin{cases} [\,s(1),\ldots,s(n-1)\,]_1 & \text{si } s(|s|) = x \\ \text{non définie} & \text{sinon.} \end{cases} \end{array}$$

Habituellement, les différentes opérations pop_x sont remplacées par une unique opération pop qui dépile le plus haut symbole de la plus haute pile de niveau 1 si elle n'est pas vide. Ces deux approches sont équivalentes si les transitions

de l'automate à pile dépendent du plus haut symbole de la pile. Cependant, l'utilisation des opérations pop_x permet de rétablir une symétrie entre push et pop et simplifie la définition des transitions d'un automate à piles de piles qui ne dépendent plus explicitement du plus haut symbole de la pile.

Opérations de niveau $k+1$. Au niveau $k+1$, la nouvelle opération est l'opération copy_k de copie de la plus haute pile de niveau k ainsi que l'opération de destruction de la plus haute pile de niveau k[1] notée destr_k. Dans ce travail, nous considérerons une version plus symétrique de la destruction, notée $\overline{\text{copy}}_k$, qui ne supprime la dernière pile de niveau k que si elle est égale à la précédente pile de niveau k. Ces opérations sont formellement définies pour tout $k \geq 1$ et $n \geq 1$ par:

$$\begin{array}{lcl} \text{copy}_k([\, s_1,\ldots,s_n\,]_{k+1}) & = & [\, s_1,\ldots,s_n,s_n\,]_{k+1} \\ \text{destr}_k([\, s_1,\ldots,s_{n+1}\,]_{k+1}) & = & [\, s_1,\ldots,s_n\,]_{k+1} \\ \overline{\text{copy}}_k([\, s_1,\ldots,s_n,s_n\,]_{k+1}) & = & [\, s_1,\ldots,s_n\,]_{k+1} \end{array}$$

Exemple 4.1.1. Considérons la pile de niveau 3 $[\,[\,[\,aab\,][\,aaa\,]\,]_2\,]_3$ et la suite d'opérations de niveau 3 ci-dessous qui la transforme:

$$\begin{array}{lll} & [\,[\,[\,aab\,][\,aaa\,]\,]_2\,]_3 & \xrightarrow{\text{copy}_2} [\,[\,[\,aab\,][\,aaa\,]\,]_2[\,[\,aab\,][\,aaa\,]\,]_2\,]_3 \\ \xrightarrow{\text{pop}_a} & [\,[\,[\,aab\,][\,aaa\,]\,]_2[\,[\,aab\,][\,aa\,]\,]_2\,]_3 & \xrightarrow{\text{push}_b} [\,[\,[\,aab\,][\,aaa\,]\,]_2[\,[\,aab\,][\,aab\,]\,]_2\,]_3 \\ \xrightarrow{\overline{\text{copy}}_1} & [\,[\,[\,aab\,][\,aaa\,]\,]_2[\,[\,aab\,]\,]_2\,]_3 & \xrightarrow{\text{destr}_2} [\,[\,[\,aab\,][\,aaa\,]\,]_2\,]_3 \end{array}$$

Si l'opération destr_2 est remplacée par l'opération $\overline{\text{copy}}_2$ alors la suite n'est plus définie car les deux dernières piles de niveau 2, $[\,[\,aab\,][\,aaa\,]\,]_2$ et $[\,[\,aab\,]\,]_2$, ne sont pas égales.

L'opération de destruction symétrique a été introduite pour des raisons techniques dans [CW03] et a été utilisée de manière systématique dans [Car05] et [Fra05]. Nous montrerons, dans le sous-paragraphe 4.1.5, que les automates à pile de piles définis en utilisant les opérations destr_k ou les opérations $\overline{\text{copy}}_k$ acceptent les mêmes langages. Nous verrons dans le chapitre 5, consacré aux graphes des automates à pile de piles, que cette équivalence est aussi valable à isomorphisme près pour les graphes de transitions associés aux automates à pile de piles (cf. théorèmes 5.1.21 et 5.1.23).

Tests de fond de pile Pour tout niveau $k \geq 1$, nous définissons une opération permettant de tester si la dernière pile de niveau k est vide.

1. Nous n'autorisons pas cette destruction dans le cas où la pile de niveau $k+1$ ne contient qu'une seule pile de niveau k, conformément à la définition des piles de niveau $k+1$.

$$T_{[\,]_k}([s_1,\ldots,s_n]_k) \;=\; \begin{cases} [\,]_k & \text{si } [s_1,\ldots,s_n]_k = [\,]_k \\ \text{non définie} & \text{sinon} \end{cases}$$

Ces opérations sont traditionnellement évitées en utilisant des encodages des piles de piles comportant des symboles de «fond de pile». Cependant, elles sont nécessaires dans la définition de la rationalité pour les piles de piles (cf. paragraphe 4.2) et nous permettent de nous abstraire de ce type d'encodage sur les piles de piles.

Monoïde des opérations de niveau k. Le jeu d'opérations symétriques de niveau au plus k sur un alphabet Γ est noté $\mathrm{Ops}_k(\Gamma)$ et est défini par:

$$\begin{array}{rcl} \mathrm{Ops}_1 & = & \{\mathrm{pop}_x,\mathrm{push}_x \mid x \in \Gamma\} \cup \{T_{[\,]_1}\} \\ \mathrm{Ops}_{k+1} & = & \mathrm{Ops}_k \cup \{\mathrm{copy}_k,\overline{\mathrm{copy}}_k\} \cup \{T_{[\,]_{k+1}}\} \end{array}$$

L'ensemble des fonctions partielles de $\mathrm{Stacks}(\Gamma)$ vers $\mathrm{Stacks}(\Gamma)$ forme un monoïde, pour la composition de fonctions, dont l'élément neutre est l'identité sur $\mathrm{Stacks}(\Gamma)$, notée Id.

Pour tout niveau $k \geq 1$, nous considérons le sous-monoïde $\mathrm{Ops}_k^\star$ défini par:

$$\mathrm{Ops}_k^\star = \{\theta = \theta_1 \cdots \theta_n \mid n \geq 1, \forall i \in [1,n], \theta_i \in \mathrm{Ops}_k \text{ et } |\theta| \geq k\}.$$

Pour la composition des fonctions, $\mathrm{Ops}_k^\star(\Gamma)$ est un monoïde dont l'élément neutre est l'identité vue comme une opération de niveau k et notée Id_k (*i.e.* Id_k est l'identité sur $\mathrm{Stacks}_{\geq k}$). La fonction vide $\emptyset$ est l'élément absorbant pour ce monoïde.

Il convient de noter que $\mathrm{Ops}_k^\star$ n'est pas le sous-monoïde engendré par Ops_k (que nous noterions Ops_k^*) car nous imposons une restriction supplémentaire sur le niveau des opérations (*i.e.* $\mathrm{Ops}^\star(\Gamma) = \{\theta \in \mathrm{Ops}_k^*(\Gamma) \mid |\theta| \geq k\}$). Ainsi, pour $x \in \Gamma$, pop_x n'appartient pas à $\mathrm{Ops}^\star$ mais pop_x vue comme une opération de niveau 2 (*i.e.* $\mathrm{Id}_2 \cdot \mathrm{pop}_x$) appartient à $\mathrm{Ops}_2^\star$.

De manière analogue, nous définissons, pour tout niveau $k \geq 1$, le jeu d'opérations classiques COps_k où les opérations $\overline{\mathrm{copy}}_\ell$ sont remplacées par les opérations destr_ℓ.

$$\begin{array}{rcl} \mathrm{COps}_1 & = & \{\mathrm{pop}_x,\mathrm{push}_x \mid x \in \Gamma\} \cup \{T_{[\,]_1}\} \\ \mathrm{COps}_{k+1} & = & \mathrm{COps}_k \cup \{\mathrm{copy}_k,\mathrm{destr}_k\} \cup \{T_{[\,]_{k+1}}\} \end{array}$$

Nous définissons de manière similaire le sous-monoïde $\mathrm{COps}_k^\star$.

4.1.3 Instructions

Les instructions permettent de travailler de manière symbolique sur le monoïde $\mathrm{Ops}_k^\star(\Gamma)$ des opérations de niveau k. À chaque opération de Ops_k, nous

associons un symbole appelé *instruction*. Nous définissons ainsi pour chaque niveau k un ensemble d'instructions Γ_k en bijection avec Ops_k:

$$\begin{array}{lcl} \Gamma_1 & = & \Gamma \cup \overline{\Gamma} \cup \{\perp_1\} \\ \Gamma_{k+1} & = & \Gamma_k \cup \{k,\bar{k}\} \cup \{\perp_k\} \end{array}$$

où $\overline{\Gamma}$ est un ensemble disjoint de Γ mais en bijection avec ce dernier. Pour tout $x \in \Gamma$, on note $\bar{x}$ l'élément correspondant dans $\overline{\Gamma}$. Dans la suite, nous noterons $\Gamma_k^{\mathrm{o}} = \Gamma \cup \bar{\Gamma} \cup \{\ell,\bar{\ell} \mid \ell \in [1,k-1]\}$ l'ensemble des instructions qui correspondent à des «modifications» de la pile et $\Gamma_k^{\mathrm{t}} = \{\perp_\ell \mid \ell \in [1,k]\}$ l'ensemble des instructions qui correspondent à des tests de fonds de piles.

Pour tout $k \geq 1$, nous définissons la bijection $\mathcal{R}_k$ de Γ_k dans Ops_k par:

$$\begin{array}{llll} \mathcal{R}_k(x) = \mathrm{push}_x & \mathcal{R}_k(\bar{x}) = \mathrm{pop}_x & & \text{pour } x \in \Gamma \\ \mathcal{R}_k(\ell) = \mathrm{copy}_\ell & \mathcal{R}_k(\bar{\ell}) = \overline{\mathrm{copy}}_\ell & & \text{pour } \ell \in [1,k-1] \\ \mathcal{R}_k(\perp_\ell) = T_{[\,]_\ell} & & & \text{pour } \ell \in [1,k] \end{array}$$

La bijection $\mathcal{R}_k$ s'étend de manière canonique en un morphisme de monoïdes entre le monoïde libre Γ_k^* et le monoïde $\mathrm{Ops}_k^\star$ des opérations de niveau k. Quand k se déduit du contexte, nous écrivons simplement $\mathcal{R}$ au lieu de $\mathcal{R}_k$. Nous dirons que $\rho \in \Gamma_k^*$ s'interprète en $\mathcal{R}_k(\rho)$. Ainsi, la suite d'instructions $a\bar{a}1b\perp_2$ s'interprète par $\mathcal{R}_2$ en $\mathrm{push}_a\mathrm{pop}_a\mathrm{copy}_1\mathrm{push}_b T_{[\,]_2} = \mathrm{copy}_1\mathrm{push}_b T_{[\,]_2} = \emptyset$.

Nous étendons la notation barrée en une application involutive de Γ_k^* dans Γ_k^* en prenant pour tout $x \in \Gamma_k^{\mathrm{o}}$, $\bar{\bar{x}} = x$ et pour tout $t \in \Gamma_k^{\mathrm{t}}$, $\bar{t} = t$. Enfin, pour tout $\rho \in \Gamma_k^*$,

$$\bar{\rho} = \overline{\rho(|\rho|)} \ldots \overline{\rho(1)}$$

Les opérations de $\mathrm{Ops}_k^\star$ sont des fonctions partielles injectives. Il suit donc que pour tout $\theta \in \mathrm{Ops}_k^\star$, la fonction partielle inverse est bien définie et sera notée θ^{-1}. Il faut remarquer que cette fonction inverse n'est pas l'inverse de θ dans le monoïde $\mathrm{Ops}_k^\star$. En effet, dès le niveau 1, il suffit de considérer l'opération pop_x qui admet pour fonction inverse l'opération push_x. Nous n'avons pas $\mathrm{pop}_x\mathrm{push}_x = \mathrm{Id}_1$.

La symétrie dans le jeu des opérations Ops_k est traduite par le lemme suivant.

Lemme 4.1.2. *Pour tout $k \geq 1$ et pour tout $\rho \in \Gamma_k^*$,*

$$\mathcal{R}(\bar{\rho}) = \mathcal{R}(\rho)^{-1}.$$

Démonstration. Soit $k \geq 1$. Il suffit de remarquer que pour tout $\gamma \in \Gamma_k$, nous avons $\mathcal{R}(\bar{\gamma}) = \mathcal{R}(\gamma)^{-1}$. La propriété découle alors d'une récurrence sur la longueur de ρ. □

Si, au lieu d'interpréter les instructions de Γ_k dans Ops_k, nous les interprétons dans COps_k, nous obtenons l'application notée $\mathcal{R}_k^{\mathrm{C}}$. Notez que $\mathcal{R}_k^{\mathrm{C}}$ ne satisfait pas de propriété analogue à celle du lemme 4.1.2.

Nous concluons par une propriété simple des suites d'instructions de niveau $k+1$ ne contenant pas d'occurrence du symbole $\bar{k}$.

Lemme 4.1.3. *Pour toute suite d'instructions ρ dans $(\Gamma_{k+1} \setminus \{\bar{k}, \perp_k\})^*$ et pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$, $\mathcal{R}(\rho)(s)$ est défini si et seulement si $\mathcal{R}(\tilde{\rho})(\mathrm{top}_k(s))$ l'est où la suite $\tilde{\rho} \in \Gamma_k^*$ est obtenue en effaçant toutes les occurrences de l'instruction k dans ρ. . De plus, si ces deux piles sont définies, $\mathrm{top}_k(\mathcal{R}(\rho)(s)) = \mathcal{R}(\tilde{\rho})(\mathrm{top}_k(s))$.*

Démonstration. La propriété est établie par une récurrence immédiate sur la longeur de ρ. □

4.1.4 Suites d'instructions réduites

Dans ce sous-paragraphe, nous introduisons la notion de suite réduite d'instructions de Γ_k qui est une suite ne contenant pas de facteur (non vide) w tel que $\mathcal{R}_k(w) \subseteq \mathrm{Id}_k$. En particulier, une telle suite ne contient pas d'instruction de tests de fonds de piles.
Ainsi, une suite réduite ρ est telle que pour toute pile s de $\mathrm{Stacks}_k(\Gamma)$, il n'existe pas deux suites $\rho' \neq \rho'' \in \Gamma_k^*$ telles que $\rho' \sqsubseteq \rho$, $\rho'' \sqsubseteq \rho$ et $\mathcal{R}(\rho')(s) = \mathcal{R}(\rho'')(s)$. Intuitivement, une suite d'instructions (quand elle est interprétée) ne revient pas sur ses pas.

Nous allons voir dans la proposition 4.1.8 que si l'on se restreint aux suites ρ telles que $\mathcal{R}(\rho) \neq \emptyset$, il suffit seulement d'interdire un nombre fini de facteurs.

Définition 4.1.4. Une suite $\rho \in \Gamma_k^*$ est *réduite* si ρ ne contient aucun facteur de la forme $t \in \Gamma_k^{\mathrm{T}}$ ni de la forme $\gamma\bar{\gamma}$ pour $\gamma \in \Gamma_k^{\mathrm{o}}$.

Nous considérerons naturellement le système de récriture $\rightarrow_k \subseteq \Gamma_k^* \times \Gamma_k^*$ défini par l'ensemble fini de règles suivantes:

$$\{(t,\varepsilon),(\gamma\bar{\gamma},\varepsilon) \mid t \in \Gamma_k^{\mathrm{T}} \text{ et } \gamma \in \Gamma_k^{\mathrm{o}}\}$$

Comme $\rightarrow_k$ est confluent et *nœthérien*, il existe, pour tout $\rho \in \Gamma_k^*$, une unique forme normale notée $\rho^{\downarrow}$ (*i.e.* $\rho \rightarrow_k^* \rho^{\downarrow}$ et $\rho^{\downarrow} \not\rightarrow_k$). La suite $\rho^{\downarrow}$ sera appelée suite réduite de ρ.

Exemple 4.1.5. La suite $\rho = \bar{a}\bar{b}1a\bar{a}\bar{1}b\bar{1}a1 \in \Gamma_2^*$ a pour suite réduite $\rho^{\downarrow} = \bar{a}\bar{1}a1$.

$$\rho \rightarrow_k \bar{a}\bar{b}1\bar{1}b\bar{1}a1 \rightarrow_k \bar{a}\bar{b}b\bar{1}a1 \rightarrow_k \bar{a}\bar{1}a1$$

Cette approche est identique à celle utilisée dans l'étude du groupe libre [Sak03]. Dans le groupe libre, cette approche est très naturelle car le système de réécriture consiste à éliminer des facteurs qui, quand ils sont interprétés dans le groupe libre,

sont égaux à l'élément neutre. Dans notre cas, cette propriété n'est pas vérifiée et le système $\rightarrow_k$ ne préserve pas l'interprétation par $\mathcal{R}_k$.

En général, $\mathcal{R}(\rho)$ est différente de $\mathcal{R}(\rho^{\downarrow})$[2]. Ainsi, dans notre exemple, la pile $\mathcal{R}(\rho)([\,[\,a\,]\,[\,aa\,]\,]_2)$ n'est pas définie alors que $\mathcal{R}(\rho^{\downarrow})([\,[\,a\,]\,[\,aa\,]\,]_2) = [\,[\,aa\,]\,[\,aa\,]\,]_2$ l'est.

Si une suite $\rho \in \Gamma_k^*$ ne s'interprète pas comme la fonction vide (*i.e.* $\mathcal{R}(\rho) \neq \emptyset$), nous pouvons préciser la forme de sa suite réduite. À cet effet, nous définissons pour tout $k \geq 1$, l'ensemble $\mathrm{Red}_k(\Gamma) \subseteq \Gamma_k^*$ comme suit:

$$\begin{cases} \mathrm{Red}_1(\Gamma) & = & \bigcup_{x \neq y \in \Gamma} \bar{\Gamma}^* \bar{x} y \Gamma^* \cup \Gamma^* \cup \bar{\Gamma}^* \\ \mathrm{Red}_{n+1}(\Gamma) & = & (\mathrm{Red}_n \bar{n})^* \mathrm{Red}'_n (n\mathrm{Red}_n)^* \cup (\mathrm{Red}_n \bar{n})^* \mathrm{Red}_n \cup \mathrm{Red}_n (n\mathrm{Red}_n)^* \end{cases}$$

où Red'_n désigne l'ensemble $\mathrm{Red}_n \setminus \{\varepsilon\}$.

Remarque 4.1.6. Pour tout $k \geq 1$, Red_k est l'ensemble des mots sur Γ_k ne contenant pas de facteur de la forme:

- $\bar{x}x$, $x\bar{x}$, $x\bar{y}$ pour $x \neq y \in \Gamma$,
- $\ell\, \Gamma_\ell^* \bar{\ell}$ ou $\bar{\ell}\ell$ pour $1 \leq \ell < k$.

Avant d'établir la proposition 4.1.8 qui justifie la définition des ensembles Red_k, nous avons besoin d'un lemme technique.

Lemme 4.1.7. *Pour toute suite $\rho \in \mathrm{Red}_k$ et pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$,*

$$s = \mathcal{R}(\rho)(s) \Rightarrow \rho = \varepsilon.$$

Démonstration. Nous procédons par récurrence sur le niveau k.
Cas de base : k=1. Soient $\rho \in \mathrm{Red}_1(\Gamma)$ et $s \in \mathrm{Stacks}_1(\Gamma)$ telles que $s = \mathcal{R}(\rho)(s)$. Il suit immédiatement que $\rho \in \Gamma^* \cup \bar{\Gamma}^*$ et donc que $\rho = \varepsilon$.
Etape de récurrence. Soient $\rho \in \mathrm{Red}_{k+1}$ et $s = [\,s_1 \ldots s_m\,]_{k+1} \in \mathrm{Stacks}_{k+1}(\Gamma)$ telles que $s = \mathcal{R}(\rho)(s)$. Par définition de Red_{k+1}, ρ est de la forme:

$$\rho = \overleftarrow{\rho_r}\bar{k} \ldots \bar{k}\overleftarrow{\rho_1}\bar{k}\overleftrightarrow{\rho}k\overrightarrow{\rho_1}k \ldots k\overrightarrow{\rho_t}$$

où $r \geq 0$, $t \geq 0$, $\overleftrightarrow{\rho} \in \mathrm{Red}_k$ et pour tout $i \in [1,r]$ et tout $j \in [1,t]$, $\overrightarrow{\rho}_i \in \mathrm{Red}_k$ et $\overleftarrow{\rho}_j \in \mathrm{Red}_k$. Comme $\mathcal{R}(\rho)(s) = s$, les entiers r et t sont égaux et appartiennent à $[0,m-1]$ (sinon le nombre de piles de niveau k de s serait différent du nombre de piles de niveau k de $\mathcal{R}(\rho)(s)$). Par définition des opérations copy_k et $\overline{\mathrm{copy}}_k$, il suit que $\mathcal{R}(\overleftrightarrow{\rho})(s_{m-r}) = s_{m-r}$. Par hypothèse de récurrence, il suit que $\overleftrightarrow{\rho} = \varepsilon$. Par définition de Red_{k+1}, il suit que $r = t = 0$ et donc que $\rho = \varepsilon$. □

2. Cependant, $\mathcal{R}(\rho)$ est toujours incluse dans $\mathcal{R}(\rho^{\downarrow})$.

Nous allons maintenant établir que les suites réduites de niveau k dont l'interprétation est non vide appartiennent à Red_k.

Proposition 4.1.8. *Pour toute suite réduite $\rho \in \Gamma_k^*$,*

$$\mathcal{R}(\rho) \neq \emptyset \Rightarrow \rho \in \mathrm{Red}_k.$$

Démonstration. Nous allons procéder par récurrence sur le niveau k.
Cas de base : k=1. Soit $\rho \in \Gamma_1^*$ une suite réduite telle que $\mathcal{R}(\rho) \neq \emptyset$. D'après la remarque 4.1.6, il suffit, pour montrer que ρ appartient à Red_1, d'établir pour tout $x \neq y \in \Gamma$ que $x\bar{y}$ n'est pas un facteur de ρ. Ceci suit du fait que $\mathcal{R}(x\bar{y}) = \emptyset$ et que $\mathcal{R}(\rho) \neq \emptyset$.
Etape de récurrence. Soit $\rho \in \Gamma_{k+1}^*$ une suite réduite telle que $\mathcal{R}(\rho) \neq \emptyset$. D'après la remarque 4.1.6 et le cas de base, il suffit, pour montrer que ρ appartient à Red_k, d'établir pour tout $\ell \in [1,k]$ que ρ ne contient aucun facteur de la forme $\ell\,\Gamma_\ell^*\overline{\ell}$. Supposons par l'absurde que $\rho = \rho_1\ell_0\rho_2\overline{\ell_0}\rho_3$ pour un certain $\ell_0 \in [1,k]$ et pour $\rho_1,\rho_3 \in \Gamma_{k+1}^*$ et $\rho_2 \in \Gamma_{\ell_0}^*$. Comme $\mathcal{R}_k(\rho) \neq \emptyset$, il suit que $\mathcal{R}_k(\ell_0\rho_2\overline{\ell_0}) \neq \emptyset$. De plus, comme $\ell_0\rho_2\overline{\ell_0} \in \Gamma_{\ell_0+1}^*$, nous avons $\mathcal{R}_{\ell_0+1}(\ell_0\rho_2\overline{\ell_0}) \neq \emptyset$. Il existe donc $s = [s_1 \dots s_m]_{\ell_0+1} \in \mathrm{Stacks}_{\ell_0+1}(\Gamma)$ telle que $s \in \mathrm{Dom}(\mathcal{R}_{\ell_0+1}(\ell_0\rho_2\overline{\ell_0}))$. Comme ρ_2 est facteur d'une suite réduite d'interprétation non vide, ρ_2 est aussi une suite réduite d'interprétation non vide. Il suit, par hypothèse de récurrence, que $\rho_2 \in \mathrm{Red}_{\ell_0}$. De plus, par définition des opérations copy_{ℓ_0} et $\overline{\mathrm{copy}}_{\ell_0}$ et comme $\rho_2 \in \Gamma_{\ell_0}^*$, il suit que $\mathcal{R}_{\ell_0}(\rho_2)(s_m) = s_m$. Il suit, par le lemme 4.1.7, que $\rho_2 = \varepsilon$. Donc $\ell_0\ell_0$ serait un facteur de ρ ce qui contredit le fait que ρ est réduite. □

Nous pouvons maintenant établir la propriété fondamentale des suites réduites qui est l'unicité de la suite réduite transformant une pile u en une pile v de même niveau.

Proposition 4.1.9. *Pour tout $k \geq 1$ et pour toutes piles $u,v \in \mathrm{Stacks}_k(\Gamma)$, il existe une unique suite réduite $\rho_{u,v} \in \Gamma_k^*$ telle que $v = \mathcal{R}(\rho_{u,v})(u)$.*

Démonstration. Nous allons commencer par établir l'existence d'une telle suite puis son unicité.

Soient $u,v \in \mathrm{Stacks}_k(\Gamma)$. Montrons qu'il existe une suite réduite ρ telle que $v = \mathcal{R}(\rho)(u)$.
Une récurrence immédiate sur le niveau établit que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, il existe une suite $\pi_s \in \Gamma_k^*$ telle que $s = \mathcal{R}(\pi_s)([\,]_k)$. Par le lemme 4.1.2, il suit que $v = \mathcal{R}(\overline{\pi}_u\pi_v)(u)$. Comme $\mathcal{R}(\overline{\pi}_u\pi_v) \subseteq \mathcal{R}((\overline{\pi}_u\pi_v)^{\downarrow})$, il suffit alors de prendre la suite réduite $(\overline{\pi}_u\pi_v)^{\downarrow}$.

Nous allons maintenant établir l'unicité de la suite réduite transformant une pile $u \in \mathrm{Stacks}_k(\Gamma)$ en une pile $v \in \mathrm{Stacks}_k(\Gamma)$. Nous procédons par récurrence sur le niveau k de ces piles.

Cas de base : k=1. Soient u,$v \in \mathrm{Stacks}_1(\Gamma)$. Soit ρ telle que $v = \mathcal{R}(\rho)(u)$. Comme $\mathcal{R}(\rho)$ est non vide, d'après la proposition 4.1.8 $\rho \in \mathrm{Red}_1$. On établit facilement que ρ doit être égale à $\overline{u_1}v_1$ où u_1,$v_1 \in \Gamma^*$ sont les uniques mots tels que $u = (u \wedge v)u_1$ et $v = (u \wedge v)v_1$ où $(u \wedge v)$ désigne le plus grand préfixe commun de u et v.

Etape de récurrence. Soient u et v deux piles de niveau $k+1$ et $\rho \in \Gamma_{k+1}^*$ une suite réduite telle que $v = \mathcal{R}(\rho)(u)$. Les piles u et v peuvent s'écrire de manière unique comme:

$$\begin{cases} u &= [\, s_1, \ldots, s_p, u_1, \ldots, u_m \,]_{k+1} \\ v &= [\, s_1, \ldots, s_p, v_1, \ldots, v_n \,]_{k+1} \end{cases}$$

où p,m et $n \geq 0$ avec $p + m \geq 1$ et $p + n \geq 1$ et $u_1 \neq v_1$ si $n > 0$ et $m > 0$.

Par la proposition 4.1.8, $\rho \in \mathrm{Red}_{n+1}(\Gamma)$ et s'écrit donc comme:

$$\rho = \overleftarrow{\rho_r}\bar{k} \ldots \bar{k}\overleftarrow{\rho_1}\bar{k}\overleftrightarrow{\rho}k\overrightarrow{\rho_1}k \ldots k\overrightarrow{\rho_t}$$

où $r \geq 0$, $t \geq 0$, $\overleftrightarrow{\rho} \in \mathrm{Red}_k$ (avec $\overleftrightarrow{\rho} \neq \varepsilon$ si $r > 0$ et $t > 0$) et pour tout $i \in [1,r]$ et tout $j \in [1,t]$, $\overrightarrow{\rho_i} \in \mathrm{Red}_k$ et $\overleftarrow{\rho_j} \in \mathrm{Red}_k$.

Dans la suite, on suppose que $n > 0$ et $m > 0$. Les autres cas sont de simples adaptations de ce qui suit.

Nous allons commencer par établir que $r = m - 1$ et $t = n - 1$.

Comme $u_1 \neq v_1$, on a nécessairement $r \geq m - 1$. Supposons par l'absurde que $r \geq m$. Comme $\mathcal{R}(\rho)(u) \neq \emptyset$, il est nécessaire que $p \geq m - r + 1$. Considérons la suite $\rho' = \overleftarrow{\rho}'\bar{k}\overleftrightarrow{\rho}k\overrightarrow{\rho}'$ où:

$$\begin{cases} \overleftarrow{\rho}' = \overrightarrow{\rho}' = \varepsilon & \text{si } r = m \\ \overleftarrow{\rho}' = \overleftarrow{\rho}_{r-m}\bar{k} \ldots \bar{k}\overleftarrow{\rho_1} \text{ et } \overrightarrow{\rho}' = \overrightarrow{\rho_1}k \ldots k\overrightarrow{\rho}_{r-m} & \text{sinon} \end{cases}$$

Comme $\mathcal{R}(\rho)(u) = v$, nous avons $\mathcal{R}(\rho')([\, s_1 \ldots s_p \,]_{n+1}) = [\, s_1 \ldots s_p \,]_{n+1}$. Comme ρ' est un facteur de la suite réduite ρ, ρ' est elle aussi une suite réduite et il suit, par le lemme 4.1.7, que $\rho' = \varepsilon$: ce qui contredit la définition de ρ'. Nous avons ainsi établi que $r = m - 1$. Comme $\mathcal{R}(\rho)(u) = v$, il suit que $t = n - 1$.

Par définition des opérations copy_n et $\overline{\mathrm{copy}}_n$, on montre que:

$$\begin{cases} u_\ell &= \mathcal{R}(\overleftarrow{\rho_\ell})(u_{\ell+1}) & \text{pour tout } \ell \in [1,m-1] \\ v_1 &= \mathcal{R}(\overleftrightarrow{\rho})(u_1) & \\ v_{\ell+1} &= \mathcal{R}(\overrightarrow{\rho_\ell})(v_\ell) & \text{pour tout } \ell \in [1,n-1] \end{cases}$$

Par hypothèse de récurrence, les suites réduites $\overleftarrow{\rho_\ell}$, $\overrightarrow{\rho_\ell}$ et $\overleftrightarrow{\rho}$ sont uniques. Il suit donc que ρ est unique. □

Un corollaire immédiat de la proposition 4.1.9 est que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, il existe une unique suite réduite $\rho_s \in \Gamma_k^*$ qui construit s à partir de la pile vide. Cette notion va jouer un rôle essentiel dans ce chapitre.

Définition 4.1.10. Pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, la *suite réduite* de s est l'unique suite réduite ρ_s dans Γ_k^* telle que $s = \mathcal{R}(\rho_s)([\,]_k)$.

Remarque 4.1.11. Pour tout niveau $k \geq 1$ et pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$, la suite réduite $\rho_s \in \Gamma_{k+1}^O$ de s ne contient pas d'occurrence de l'instruction $\bar{k}$. En effet, par la proposition 4.1.8, ρ_s appartient à Red_{k+1}. Si l'on suppose, par l'absurde, que ρ_s contient une occurrence de $\bar{k}$, il suit de la définition de Red_{k+1} que ρ_s a un préfixe ρ' dans $\Gamma_k^*\bar{k}$. Or par définition de $\overline{\mathrm{copy}}_k$, la pile vide $[\,]_{k+1}$ n'appartient pas à $\mathrm{Dom}(\rho')$ et donc pas à $\mathrm{Dom}(\rho_s)$.

De plus, nous définissons, pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, $\mathrm{Last}(s) \in \Gamma_k^o \cup \{\, k, \varepsilon \,\}$ par:

$$\mathrm{Last}(s) = \begin{cases} \rho_s(|\rho_s|) & \text{si } s \neq [\,]_k, \\ \varepsilon & \text{sinon.} \end{cases}$$

Une autre propriété de la suite réduite d'une pile $s \in \mathrm{Stacks}_k(\Gamma)$ est d'être la plus petite suite $\rho \in \Gamma_k^*$ telle que $s = \mathcal{R}(\rho)([\,]_k)$. Donc par la proposition 4.1.9, la plus petite suite d'instructions, construisant une pile donnée à partir de la pile vide, est unique. Cette propriété n'est plus vérifiée lorsque l'on considère le jeu d'opérations classiques comme cela a déjà été remarqué dans [Wöh05]. Ainsi si nous considérons la pile $s = [\,[\,[\,a\,]\,[\,aaa\,]\,]_2\,[\,[\,a\,]\,[\,a\,]\,]_2\,]_3$ de niveau 3 dont la suite réduite est $a1aa2\bar{a}\bar{a}$ qui est de longueur 7. Avec le jeu d'opérations classiques, il existe deux suites d'opérations de COps_3 de longueur 7 qui construisent s à partir de la pile vide $[\,]_3$:

$$\begin{cases} s &= \mathrm{push}_a\mathrm{copy}_1\mathrm{push}_a\mathrm{push}_a\mathrm{copy}_2\mathrm{pop}_a\mathrm{pop}_a([\,]_3) \\ s &= \mathrm{push}_a\mathrm{copy}_1\mathrm{push}_a\mathrm{push}_a\mathrm{copy}_2\mathrm{destr}_1\mathrm{copy}_1([\,]_3) \end{cases}$$

Il est raisonnable de penser qu'il n'existe pas de notion «canonique» de suite d'opérations classiques associée à une pile de niveau k. Nous verrons, dans le paragraphe 4.6, que cette différence a des conséquences sur la notion d'ensemble rationnel de piles de niveau k associée aux opérations classiques (cf. paragraphe 4.6).

Nous allons conclure par un lemme technique qui décrit la structure des suites non réduites $\rho \in \Gamma_k^*$ telles que pour une certaine pile $s \in \mathrm{Stacks}_k(\Gamma)$, $s = \mathcal{R}(\rho)(s)$.

Lemme 4.1.12. *Pour toute suite non vide $\rho \in (\Gamma_k^o)^*$ et pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ telles que $s = \mathcal{R}(\rho)(s)$, la suite ρ se décompose de manière unique sous la forme $\rho_1, \ldots, \rho_n$ avec pour tout $i \in [1,n]$, $\rho_i \in (\Gamma_k^o)^+$, $\mathcal{R}(\rho_1 \ldots \rho_i)(s) = s$ et pour tout $\rho' \sqsubseteq \rho$, si $\mathcal{R}(\rho')(s) = s$ alors $\rho' \in \{\rho_1 \ldots \rho_i \mid i \in [0,n]\}$.*
De plus, pour tout $i \in [1,n]$, $\rho_i(1) = \overline{\rho_i(|\rho_i|)}$.

Démonstration. L'existence et l'unicité d'une telle décomposition sont immédiates. Pour la dernière partie de la proposition, il suffit d'établir que pour toute

suite $\rho \in (\Gamma_k^o)^+$ telle que $\mathcal{R}(\rho)(s) = s$ pour une pile $s \in \text{Stacks}_k(\Gamma)$ et telle que pour tout $\rho' \neq \varepsilon \sqsubset \rho$, $\mathcal{R}(\rho)(s) \neq s$, nous avons $\rho(1) = \overline{\rho(|\rho|)}$.

Si l'on note $\gamma = \rho(1)$ alors $\rho = \gamma\rho'$ où $\rho' \in (\Gamma_k^o)^+$. Par la proposition 4.1.9, $\rho^\downarrow = \varepsilon$ et par confluence de $\to_k$, $(\rho')^\downarrow = \bar{\gamma}$. La suite ρ' peut donc s'écrire comme $\rho_1\bar{\gamma}\rho_2$ où ρ_1,$\rho_2 \in (\Gamma_k^o)^*$ et $\rho_1^\downarrow = \rho_2^\downarrow = \varepsilon$. Pour conclure, il suffit de remarquer que $\rho_2 = \varepsilon$. En effet, si ce n'est pas le cas $\gamma\rho_1\bar{\gamma} \sqsubset \rho$ et $\mathcal{R}(\gamma\rho_1\bar{\gamma})(s) = s$. □

4.1.5 Automates à pile de piles et leurs langages

Dans ce sous-paragraphe, nous présentons formellement les automates à pile de niveau k sur le jeu d'opérations symétriques Ops_k et sur le jeu d'opérations classiques COps_k. Nous donnons à cette occasion un bref historique de la notion d'automate à pile de piles. Enfin, nous établissons l'équivalence entre les deux modèles d'automates à chaque niveau en tant qu'accepteurs de langages. Ce résultat a été obtenu dans [CW03] en collaboration avec Stefan Wöhrle. La preuve complète apparait dans [Wöh05]. Indépendamment, ce résultat a été obtenu dans [Fra05].

Définition 4.1.13. Un automate à pile $\mathcal{A}$ de niveau k sur le jeu d'opérations Ops_k est donné par un septuplet $(\Gamma,\Sigma,\tau,Q,I,F,\Delta)$ où:

- Γ et Σ sont respectivement les alphabets finis de pile et des étiquettes,
- $\tau \in \Sigma$ est une étiquette jouant le rôle d'action silencieuse,
- Q est un ensemble fini d'états,
- $I \subseteq Q$ et $F \subseteq Q$ sont respectivement les ensembles d'états initiaux et finaux,
- et $\Delta \subseteq Q \times \Sigma \times \text{Ops}_k^\star \times Q$ est l'ensemble des transitions.

Une transition $(p,a,\theta,q) \in \Delta$ sera notée $p \xrightarrow{a} (q,\theta)$. Une *configuration* de $\mathcal{A}$ est un couple dans $Q \times \text{Stacks}_k(\Gamma)$. L'ensemble des configurations initiales (resp. finales) est $I \times \text{Stacks}_k(\Gamma)$ (resp. $F \times \text{Stacks}_k(\Gamma)$). Pour tout $a \in \Sigma$, l'automate $\mathcal{A}$ induit une relation $\xrightarrow[\mathcal{A}]{a}$ sur ses configurations définie par:

$$(p,w) \xrightarrow[\mathcal{A}]{a} (q,w') \Leftrightarrow \exists (p,a,\theta,q) \in \Delta, w' = \theta(w).$$

Cette relation induit pour tout $u \in (\Sigma \setminus \{\tau\})^*$ une relation $\overset{u}{\underset{\mathcal{A}}{\Longrightarrow}}$ définie par:

$$\begin{array}{rcl} \overset{\varepsilon}{\underset{\mathcal{A}}{\Longrightarrow}} & = & (\xrightarrow[\mathcal{A}]{\tau})^* \\ \overset{ua}{\underset{\mathcal{A}}{\Longrightarrow}} & = & \overset{u}{\underset{\mathcal{A}}{\Longrightarrow}} \cdot \xrightarrow[\mathcal{A}]{a} \cdot (\xrightarrow[\mathcal{A}]{\tau})^* \end{array}$$

où $u \in (\Sigma \setminus \{\tau\})^*$ et $a \in \Sigma \setminus \{\tau\}$.

Le *langage accepté* par l'automate à pile $\mathcal{A}$, noté $\mathcal{L}(\mathcal{A})$, est l'ensemble des mots $u \in (\Sigma \setminus \{\tau\})^*$ pour lesquels il existe $i \in I$, $f \in F$ et $s \in \mathrm{Stacks}_k(\Gamma)$ tels que $(i, [\,]_k) \xRightarrow[\mathcal{A}]{u} (f,s)$.

Exemple 4.1.14. Nous définissons un automate $\mathcal{A} = (\Gamma,\Sigma,\tau,Q,I,F,\Delta)$ à pile de niveau 2 avec $\Sigma = \{a,b,\$,\tau\}$ et $\Gamma = \{a,b\}$ qui accepte le langage:

$$L = \{w\$w \mid w \in \{a,b\}^*\}.$$

L'ensemble des états Q est $\{i,p,q,f\}$ où i est l'unique état initial et f est l'unique état final. L'ensemble des transitions Δ est donné par:

$$\begin{array}{lll}
i \xrightarrow{a} (i,\mathrm{push}_a) & i \xrightarrow{b} (i,\mathrm{push}_b) & i \xrightarrow{\$} (p,\mathrm{copy}_1) \\
p \xrightarrow{\tau} (p,\mathrm{copy}_1\mathrm{pop}_a) & p \xrightarrow{\tau} (p,\mathrm{copy}_1\mathrm{pop}_b) & p \xrightarrow{\tau} (q,\bot_1) \\
q \xrightarrow{a} (q,\mathrm{push}_a\overline{\mathrm{copy}}_1) & q \xrightarrow{b} (q,\mathrm{push}_b\overline{\mathrm{copy}}_1) & q \xrightarrow{\tau} (f,\overline{\mathrm{copy}}_1)
\end{array}$$

Le mot $abb\$abb$ est accepté par le calcul:

$$\begin{array}{l}
(i, [\,[\,abb\,]\,]_2) \xrightarrow[\mathcal{A}]{abb} (i, [\,[\,[\,abb\,]\,]\,]_2) \xrightarrow[\mathcal{A}]{\$} (p, [\,[\,[\,abb\,]\,]\,[\,abb\,]\,]_2) \xrightarrow[\mathcal{A}]{\tau^4} (q, [\,[\,[\,abb\,]\,]\,[\,abb\,]\,\ldots\,[\,]\,]_2) \\
\xrightarrow[\mathcal{A}]{a} (q, [\,[\,[\,abb\,]\,]\,[\,abb\,]\,[\,ab\,]\,[\,a\,]\,]_2) \xrightarrow[\mathcal{A}]{b} (q, [\,[\,[\,abb\,]\,]\,[\,abb\,]\,[\,ab\,]\,]_2) \\
\xrightarrow[\mathcal{A}]{b} (q, [\,[\,[\,abb\,]\,]\,[\,abb\,]\,]_2) \xrightarrow[\mathcal{A}]{\tau} (f, [\,[\,[\,abb\,]\,]\,]_2)
\end{array}$$

Remarque 4.1.15. Si nous ne considérons que le langage accepté, nous pouvons supposer, sans perte de généralité, que les transitions[3] de l'automate à pile sont dans $\Delta \subseteq Q \times \Sigma \times \mathrm{Ops}_k \times Q$.

Si nous remplaçons $\mathrm{Ops}_k^\star$ par $\mathrm{COps}_k^\star$ dans la définition 4.1.13, nous obtenons la notion d'automate à pile sur COps_k qui est la notion classique d'automate à pile de niveau k.

Les automates à pile d'ordre supérieur ont été introduits dans les années 70. Dans [Gre70], Greibach attribue l'idée de ces automates à Aho et Ullman. Ces automates sont aussi définis dans [Mas76]. Les automates sur COps_k diffèrent légèrement des automates considérés par ces auteurs et de ceux considérés dans [DG86, Eng91]. La différence majeure réside dans la définition des piles de piles. Les piles de niveau k sont des suites non vides de couples formés d'un symbole de pile et d'une pile de niveau $k-1$. Comme remarqué dans [KNU02], ces symboles de pile supplémentaires peuvent être simulés dans le modèle des automates sur COps_k.

Les langages acceptées par les automates à pile sur COps_k sont connus sous le nom de langages k-OI ou de langages indexés de niveau k. La première dénomination vient des travaux de Damm [Dam82] où ces langages sont introduits

3. Il faudrait en toute rigueur remplacer Ops_k par $\mathrm{Id}_k \cdot \mathrm{Ops}_k$.

en utilisant des OI macro-grammaires de niveau k. L'équivalence entre cette présentation et le formalisme des automates à pile d'ordre supérieurs est montrée dans [DG86]. La deuxième dénomination vient de la notion de grammaire indexée (de niveau 2) introduite par Aho dans [Aho68] et de sa généralisation à tout niveau par Maslov dans [Mas74]. Pour le niveau 2, l'équivalence entre les grammaires indexées (de niveau 2) et les langages acceptés par les automates à pile sur COps_2 est obtenue dans [Aho69]. A tout niveau, cette équivalence a été établie dans [Mas76]. Pour une présentation détaillée sur la notion d'automates à pile de piles, nous référons le lecteur à [Eng91].

Dans [Eng91], l'auteur établit que les langages acceptés par les automates à pile sur COps_k sont strictement inclus dans les langages acceptés par les automates à piles sur COps_{k+1}. Il suit donc que la hiérarchie des langages indexées d'ordre supérieur est stricte.

Théorème 4.1.16 ([Eng91])**.** *Pour tout $k \geq 1$, les langages indexés de niveau k sont strictement inclus dans les langages indexés de niveau $k+1$.*

Dans le même article [Eng91], la complexité du test du vide du langage accepté par un automate à piles sur COps_k est étudiée et l'auteur donne une borne inférieure et une borne supérieure qui sont rappelées dans le théorème suivant.

Théorème 4.1.17 ([Eng91])**.** *Le problème du test du vide des langages acceptés par les automates à pile sur* COps_k *est complet pour les réductions en espace logarithmique pour la classe* $\bigcup_{d>1} DTIME(2^{\uparrow^{k-1}(dn^2)})$.

La fin de ce sous-paragraphe est dédiée à établir l'équivalence, du point de vue des langages acceptés, entre les automates à pile de niveau $k \geq 1$ sur Ops_k et sur COps_k.

4.1.5.1 Des automates sur COps_k aux automates sur Ops_k

Pour transformer un automate à piles sur COps_k en un automate à pile sur Ops_k, il suffit de remarquer que pour tout $\ell < k$,

$$\mathrm{destr}_\ell = \mathrm{Ops}_\ell^* \cdot \overline{\mathrm{copy}}_\ell. \tag{4.1}$$

Proposition 4.1.18. *Tout langage accepté par un automate à pile sur* COps_k *est accepté par un automate à pile sur* Ops_k *de même niveau.*

Démonstration. Soit $\mathcal{A} = (\Gamma,\Sigma,\tau,Q,I,F,\Delta)$ un automate à pile sur COps_k. Par la remarque 4.1.15, nous pouvons supposer sans perte de généralité que $\Delta \subseteq Q \times \Sigma \times \mathrm{Ops}_k \times Q$. Pour tout $\ell \in [1,k-1]$, nous définissons l'ensemble $\Delta_\ell = \{\delta \in \Delta \mid \delta = (p,x,\mathrm{destr}_\ell,q)\}$ des transitions avec pour opération destr_ℓ. Pour chaque $\ell \in [1,k-1]$, soit Q_ℓ un ensemble en bijection avec Δ_ℓ. Pour toute transition

$\delta \in \Delta_\ell$, nous noterons q_δ l'état correspondant de Q_ℓ. De plus, nous supposons que les Q_ℓ sont deux à deux disjoints et sont disjoints de Q.

Nous pouvons maintenant définir un automate $\mathcal{B} = (\Gamma,\Sigma,\tau,Q_\mathcal{B},I_\mathcal{B},F_\mathcal{B},\Delta_\mathcal{B})$ à pile sur Ops_k où $Q_\mathcal{B} = Q \cup \bigcup_{\ell\in[1,k-1]} Q_\ell$, $I_\mathcal{B} = I$, $F_\mathcal{B} = F$ et où:

$$\begin{array}{rcl}
\Delta_\mathcal{B} & = & \Delta \setminus \bigcup_{\ell\in[1,k-1]} \Delta_\ell \\
& \cup & \{(p,\tau,\mathrm{Id}_k,q_\delta) \mid \delta = (p,x,\theta,q) \in \bigcup_{\ell\in[1,k-1]} \Delta_\ell\} \\
& \cup & \bigcup_{\ell\in[1,k-1]}\{(q_\delta,\tau,\theta,q_\delta) \mid \delta \in \Delta_\ell \text{ et } \theta \in \mathrm{Ops}_\ell\} \\
& \cup & \bigcup_{\ell\in[1,k-1]}\{(q_\delta,x,\overline{\mathrm{copy}}_\ell,p) \mid \delta = (q,x,\theta,p) \in \Delta_\ell\}
\end{array}$$

Par l'équation (4.1), il suit que $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$. □

4.1.5.2 Des automates sur Ops_k aux automates sur COps_k

L'inclusion réciproque, plus délicate, utilise un encodage des piles de niveau k. À chaque pile de niveau k sur un alphabet Γ, nous allons associer sa version «encodée», notée $[\![s]\!]$, qui est une pile de même niveau sur l'alphabet Γ_k°. Cet encodage permet de faire «apparaître» explicitement la suite réduite de s dans la pile encodée $[\![s]\!]$. Nous définissons ensuite pour chaque $\theta \in \mathrm{Ops}_k$ un sous-ensemble fini $[\![\theta]\!]_k$ de $\mathrm{COps}_k^\star$ simulant θ (*i.e.* $[\![\theta]\!]_k([\![s]\!]) = [\![\theta(s)]\!]$).

À partir de maintenant, nous fixons l'alphabet de pile Γ. Nous commençons par définir par récurrence sur le niveau k, une application $[\![\cdot]\!]$ de $\mathrm{Stacks}_k(\Gamma)$ dans $\mathrm{Stacks}_k(\Gamma_k^\circ)$.

Au niveau 1, l'encodage est simplement l'identité. Pour tout $s \in \mathrm{Stacks}_1(\Gamma)$, $[\![s]\!] = s$.

Au niveau $k+1 \geq 2$, considérons une pile $s \neq [\,]_{k+1}$ avec sa suite réduite $\rho \in (\Gamma_k^\circ \cup \{k\})^*$ (cf. remarque 4.1.11). Pour tout $\ell \in [1,|\rho|]$, nous définissons $\rho_\ell = \rho(1)\dots\rho(\ell)$ et $\tilde{\rho}_\ell$ la suite obtenue en effaçant les occurrences de l'instruction k dans ρ_ℓ. La suite $\tilde{\rho}_\ell$ appartient à $(\Gamma_k^\circ)^*$. Enfin, nous définissons, pour tout $\ell \in [1,|\rho|]$, la pile $s_\ell = \mathcal{R}(\tilde{\rho}_\ell)([\,]_k)$.

Nous pouvons maintenant définir $[\![\cdot]\!]$ pour les piles de niveau $k+1$.

$$\begin{array}{rcl}
[\![[\,]_{k+1}]\!] & = & [[\![[\,]_k]\!]]_{k+1} \\
[\![s]\!] & = & [\,[\![[\,]_k]\!],\mathrm{push}_{\rho(1)}([\![s_1]\!]),\dots,\mathrm{push}_{\rho(|\rho|)}([\![s_{|\rho|}]\!])\,]_{k+1}
\end{array} \tag{4.2}$$

Comme par la proposition 4.1.9 la suite minimale associée à une pile de piles est unique, l'application $[\![\cdot]\!]$ est une injection.

Remarque 4.1.19. Par définition de $s_{|\rho|}$ et par le lemme 4.1.3, la pile $s_{|\rho|}$ est égale à la dernière pile de niveau k de s.

Exemple 4.1.20. Considérons la pile de niveau 2 $[\,[\,abc\,]\,[\,ad\,]\,]_2$. Sa suite réduite est $\rho = abc1\bar{c}\bar{b}d$. Les piles de niveau 1 $s_1,\dots,s_7$ intervenant dans la définition de

$[\![s]\!]$ sont:

$$\begin{array}{llll} s_1 = [\, a \,]_1 & s_3 = [\, abc \,]_1 & s_5 = [\, ab \,]_1 & s_7 = [\, ad \,]_1 \\ s_2 = [\, ab \,]_1 & s_4 = [\, abc \,]_1 & s_6 = [\, a \,]_1 \,. & \end{array}$$

Donc l'encodage de $[\,[\, abc \,][\, ad \,]\,]_2$ est:

$$[\![\ [\,[\, abc \,][\, ad \,]\,]_2\ ]\!] = \left[\,[\,][\, aa \,][\, abb \,][\, abcc \,][\, abc1 \,][\, ab\bar{c} \,]\left[\, a\bar{b} \,\right][\, add \,]\,\right]_2 .$$

Pour tout niveau k et pour toute opération $\theta \in \mathrm{Ops}_k$, nous allons définir un sous-ensemble fini $[\![\, \theta \,]\!]_k$ de $\mathrm{COps}_k^\star$ qui va «simuler» le comportement de θ sur l'encodage d'une pile de niveau k. Ces ensembles auront la propriété supplémentaire d'être *non-ambigus*. Un sous-ensemble fini $R \subseteq \mathrm{COps}_k^\star$ est non-ambigu si pour tout $\theta \neq \theta' \in R$, $\mathrm{Dom}(\theta) \cap \mathrm{Dom}(\theta') = \emptyset$. Il suit donc que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, $R(s)$ est soit vide soit un singleton. Nous utiliserons donc la notation fonctionnelle et dirons que $R(s)$ n'est pas défini si $R(s) = \emptyset$ et que $R(s) = s'$ si $R(s) = \{\, s' \,\}$.

Notre but sera donc d'établir (cf. lemme 4.1.21) que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ et pour toute opération $\theta \in \mathrm{Ops}_k$, $[\![\, \theta \,]\!]_k([\![\, s \,]\!])$ est défini si et seulement si $\theta(s)$ l'est et dans ce cas, $[\![\, \theta \,]\!]_k([\![\, s \,]\!]) = [\![\, \theta(s) \,]\!]$.

Comme pour l'encodage des piles, l'encodage des opérations est défini par récurrence sur le niveau k[4].

Au niveau 1, nous prenons pour tout $\theta \in \mathrm{Ops}_1$

$$[\![\, \theta \,]\!]_1 = \{\theta\}.$$

Au niveau $k+1 \geq 2$, nous définissons pour tout $\theta \in \mathrm{Ops}_k$, l'encodage comme suit:

$$\begin{array}{rcll} [\![\, \theta \,]\!]_{k+1} & = & \bigcup_{\gamma \neq \overline{\gamma_\theta} \in \Gamma^\circ_{k+1}} \mathrm{pop}_\gamma \mathrm{push}_\gamma \mathrm{copy}_k \mathrm{pop}_\gamma [\![\, \theta \,]\!]_k \mathrm{push}_{\gamma_\theta} & \text{(a)} \\ & \cup & \{\mathrm{pop}_{\overline{\gamma_\theta}} \mathrm{destr}_k\} & \text{(b)} \\ & \cup & T_{[\,]_{k+1}} \mathrm{copy}_k [\![\, \theta \,]\!]_k \mathrm{push}_{\gamma_\theta} & \text{(c)} \end{array} \tag{4.3}$$

où $\gamma_\theta = \mathcal{R}^{-1}(\theta)$ est l'instruction de Γ°_{k+1} correspondant à θ.

$$\begin{aligned} [\![\, \mathrm{copy}_k \,]\!]_{k+1} &= \bigcup_{\gamma \in \Gamma^\circ_{k+1}} \mathrm{pop}_\gamma \mathrm{push}_\gamma \mathrm{copy}_k \mathrm{pop}_\gamma \mathrm{push}_k \\ &\qquad \cup\ \{T_{[\,]_{k+1}} \mathrm{copy}_k \mathrm{push}_k\} \end{aligned} \tag{4.4}$$

$$[\![\, \overline{\mathrm{copy}}_k \,]\!]_{k+1} = \{\mathrm{pop}_k \mathrm{destr}_k\} \tag{4.5}$$

$$[\![\, T_{[\,]_{k+1}} \,]\!]_{k+1} = \{T_{[\,]_{k+1}}\}. \tag{4.6}$$

4. Dans un souci de clarté, pour tout $\theta \in \mathrm{COps}_k^\star$ et $R \subset \mathrm{COps}_k^\star$, nous écrirons simplement $\theta \cdot R$ au lieu de $\{\theta\} \cdot R$.

L'ensemble $[\![\theta]\!]_{k+1}$ est bien non-ambigu car les différentes opérations qui le composent commencent par dépiler des symboles différents ou par tester que la pile est vide. Il en va de même pour $[\![\mathrm{copy}_k]\!]_{k+1}$, $[\![\overline{\mathrm{copy}}_k]\!]_{k+1}$ et $[\![T_{[\,]_{k+1}}]\!]_{k+1}$.

Nous allons maintenant établir que les encodages des opérations simulent bien les opérations sur les encodages des piles.

Lemme 4.1.21. *Pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ et pour tout $\theta \in \mathrm{Ops}_k$,*

$$[\![\theta]\!]_k([\![s]\!]) = \begin{cases} [\![\theta(s)]\!] & \text{si } \theta(s) \text{ est défini,} \\ \text{non définie} & \text{sinon.} \end{cases}$$

Démonstration. Nous procédons par récurrence sur le niveau k.
Cas de base : $k = 1$. La propriété est immédiate.
Etape de récurrence. Soit s une pile dans $\mathrm{Stacks}_{k+1}(\Gamma)$. Nous allons distinguer deux cas selon que s est vide ou non.
Cas $s = [\,]_{k+1}$. Pour les opérations copy_k, $\overline{\mathrm{copy}}_k$ et $T_{[\,]_{k+1}}$, la propriété découle directement de la définition de l'encodage de ces opérations (cf. équations (4.4), (4.5) et (4.6)).

Considérons maintenant le cas où $\theta \in \mathrm{Ops}_k$.

	$[\![\theta]\!]_{k+1}([\![\,[\,]_{k+1}]\!])$ est défini	
$\Leftrightarrow$	$T_{[\,]_{k+1}}\mathrm{copy}_k[\![\theta]\!]_k\mathrm{push}_{\gamma_\theta}([\,]_{k+1})$ l'est	par Eq. (4.3.a)
$\Leftrightarrow$	$[\![\theta]\!]_k([\![\,[\,]_k]\!])$ l'est	car $[\![\theta]\!]_k \subseteq \mathrm{Ops}_k^\star$
$\Leftrightarrow$	$\theta([\,]_k)$ l'est	par HR
$\Leftrightarrow$	$\theta([\,]_{k+1})$ l'est	comme $\theta \in \mathrm{Ops}_k$

Supposons que $s' = \theta([\,]_{k+1})$ est définie. Par la proposition 4.1.9, la suite réduite ρ' de s' est $\mathcal{R}^{-1}(\theta) = \gamma_\theta$.

$$\begin{array}{rcll} [\![\theta]\!]_{k+1}([\![\,[\,]_{k+1}]\!]) & = & T_{[\,]_{k+1}}\mathrm{copy}_k[\![\theta]\!]_k\mathrm{push}_{\gamma_\theta}([\,]_{k+1}) & \text{par Eq. (4.3.c)} \\ & = & \big[\,[\,]_k\,,\mathrm{push}_{\gamma_\theta}([\![\theta]\!]_k([\,]_k))\,\big]_{k+1} & \\ & = & \big[\,[\,]_k\,,\mathrm{push}_{\gamma_\theta}([\![\theta([\,]_k)]\!])\,\big]_{k+1} & \text{par HR} \\ & = & [\![s']\!] & \text{par Eq. (4.2)} \\ & = & [\![\theta([\,]_{k+1})]\!] & \end{array}$$

Cas $s = [\,s_1 \dots s_m\,] \neq [\,]_{k+1}$.

La suite réduite ρ de s est non vide.

$$[\![s]\!] = \big[\,[\![\,[\,]_k]\!],\mathrm{push}_{\rho(1)}([\![s_1]\!]),\dots,\mathrm{push}_{\rho(|\rho|)}([\![s_{|\rho|}]\!])\,\big]_{k+1}.$$

Soit $\theta \in \mathrm{Ops}_{k+1}$. Nous allons distinguer plusieurs cas.

Sous-cas $\theta \in \{\mathrm{copy}_k, \overline{\mathrm{copy}}_k, T_{[\,]_{k+1}}\}$.

Nous allons établir le cas le plus intéressant $\theta = \overline{\mathrm{copy}}_k$. Les deux autres cas sont similaires.

Supposons que $[\![\overline{\mathrm{copy}}_k]\!]_{k+1}([\![s]\!])$ soit défini et montrons que $\overline{\mathrm{copy}}_k(s)$ l'est aussi. Par définition de l'encodage, il suit que $\rho(|\rho|) = k$. Donc $\overline{\mathrm{copy}}_k(s)$ a pour suite réduite $\rho(1)\dots\rho(|\rho|-1) \sqsubset \rho$ et donc est bien défini.

Supposons que $\overline{\mathrm{copy}}_k(s)$ soit définie. Nous avons $\rho(|\rho|) = k$ car sinon $\overline{\mathrm{copy}}_k(s)$ aurait pour suite réduite $\rho\bar{k}$. Or, par la remarque 4.1.11, la suite réduite d'une pile de niveau $k+1$ ne peut contenir d'occurrence de l'instruction $\bar{k}$. Donc $\overline{\mathrm{copy}}_k(s)$ a pour suite réduite $\rho(1)\dots\rho(|\rho|-1)$.

$$\begin{aligned}
[\![\overline{\mathrm{copy}}_k]\!]_{k+1}([\![s]\!]) &= \mathrm{pop}_k \mathrm{destr}_k([\![s]\!]) \\
&= [\,[\![[\,]_k]\!], \mathrm{push}_{\rho(1)}([\![s_1]\!]), \dots, \mathrm{push}_{\rho(|\rho|-1)}([\![s_{|\rho|-1}]\!])\,]_{k+1} \\
&= [\![\overline{\mathrm{copy}}_k(s)]\!]
\end{aligned}$$

Sous-cas $\theta \in \mathrm{Ops}_k \setminus \{\, \mathcal{R}(\overline{\rho(|\rho|)}) \,\}$.

Nous commençons par montrer que $[\![\theta]\!]_{k+1}([\![s]\!])$ est défini si et seulement si $\theta(s)$ l'est.

$$\begin{aligned}
& [\![\theta]\!]_{k+1}([\![s]\!]) \text{ est défini} \\
\Leftrightarrow\ & \mathrm{pop}_{\rho(|\rho|)}\mathrm{push}_{\rho(|\rho|)}\mathrm{copy}_k\mathrm{pop}_{\rho(|\rho|)}[\![\theta]\!]_k\mathrm{push}_{\gamma_\theta}([\![s]\!]) \text{ l'est} && \text{par Eq. (4.3.a)} \\
\Leftrightarrow\ & [\![\theta]\!]_k([\![s_{\rho(|\rho|)}]\!]) \text{ l'est} \\
\Leftrightarrow\ & [\![\theta]\!]_k([\![s_m]\!]) \text{ l'est} && \text{par Rem. 4.1.19} \\
\Leftrightarrow\ & \theta(s_m) \text{ l'est} && \text{par HR} \\
\Leftrightarrow\ & \theta(s) \text{ l'est} && \text{car } \theta \in \mathrm{Ops}_k
\end{aligned}$$

Dans le cas où les deux sont définis, comme γ_θ est différent de $\overline{\rho(|\rho|)}$, il suit que la suite réduite de $\theta(s)$ est $\rho\gamma_\theta$ et donc l'encodage de $\theta(s)$ est égal à

$$[\![\theta(s)]\!] = \big[\,[\![[\,]_k]\!], \mathrm{push}_{\rho(1)}([\![s_1]\!]), \dots, \mathrm{push}_{\rho(|\rho|)}([\![s_{|\rho|}]\!])\mathrm{push}_{\gamma_\theta}([\![\theta(s_{|\rho|})]\!])\,\big]_{k+1}$$

Nous avons donc:

$$\begin{aligned}
& [\![\theta]\!]_{k+1}([\![s]\!]) \\
=\ & \mathrm{pop}_{\rho(|\rho|)}\mathrm{push}_{\rho(|\rho|)}\mathrm{copy}_k\mathrm{pop}_{\rho(|\rho|)}[\![\theta]\!]_k\mathrm{push}_{\gamma_\theta}([\![s]\!]) \\
=\ & \big[\,[\![[\,]_k]\!], \mathrm{push}_{\rho(1)}([\![s_1]\!]), \dots, \mathrm{push}_{\rho(|\rho|)}([\![s_{|\rho|}]\!])\mathrm{push}_{\gamma_\theta}([\![\theta]\!]_k([\![s_{|\rho|}]\!]))\,\big]_{k+1} \\
=\ & \big[\,[\![[\,]_k]\!], \mathrm{push}_{\rho(1)}([\![s_1]\!]), \dots, \mathrm{push}_{\rho(|\rho|)}([\![s_{|\rho|}]\!])\mathrm{push}_{\gamma_\theta}([\![\theta(s_{|\rho|})]\!])\,\big]_{k+1} \\
=\ & [\![\theta(s)]\!]
\end{aligned}$$

Sous-cas $\theta = \mathcal{R}(\overline{\rho(|\rho|)})$.

Dans ce cas $\theta(s)$ est défini et a pour suite réduite $\rho(1)\dots\rho(|\rho|-1)$ par la proposition 4.1.9. L'encodage de $\theta(s)$ est donc:

$$[\![\theta(s)]\!] = \big[\,[\![[\,]_k]\!], \mathrm{push}_{\rho(1)}([\![s_1]\!]), \dots, \mathrm{push}_{\rho(|\rho|-1)}([\![s_{|\rho|-1}]\!])\,\big]_{k+1}.$$

D'après les équations (4.3.b) et (4.2), $[\![\theta]\!]_{k+1}([\![s]\!])$ est défini et égal à $[\![\theta s]\!]$. □

Nous étendons de manière canonique l'encodage $[\![\cdot]\!]_k$ de Ops_k à $\mathrm{Ops}_k^\star$.

Proposition 4.1.22. *Tout langage accepté par un automate à pile sur* Ops_k *est accepté par un automate à pile sur* COps_k.

Démonstration. Soit $\mathcal{A} = (\Gamma,\Sigma,\tau,Q,I,F,\Delta)$ un automate à pile sur $\mathrm{Ops}_k^\star$. Nous définissons un automate $\mathcal{B} = (\Gamma,\Sigma,\tau,Q,I,F,\Delta_\mathcal{B})$ où l'ensemble des transitions $\Delta_\mathcal{B}$ est défini par:

$$\Delta_\mathcal{B} = \{(p,x,\eta,q) \mid (p,x,\theta,q) \in \Delta \text{ et } \eta \in [\![\, \theta \,]\!]_k\}.$$

Par le lemme 4.1.21, il suit que pour tout $w \in (\Sigma)^*$, $p \in Q$ et $s \in \mathrm{Stacks}_k(\Gamma)$:

$$(q_0, [\ ]_k) \xrightarrow[\mathcal{A}]{w} (p,s) \quad \Leftrightarrow \quad (q_0, [\ ]_k) \xrightarrow[\mathcal{A}]{w} (p, [\![\, s \,]\!]).$$

Donc, nous avons $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$. □

En combinant les propositions 4.1.22 et 4.1.18, nous obtenons l'équivalence précédemment annoncée.

Théorème 4.1.23. *Les langages acceptés par les automates à pile sur* Ops_k *et sur* COps_k *coïncident.*

Ce résultat sera affiné dans le Chapitre 5 par les théorèmes 5.1.21 et 5.1.23 qui étendent cette équivalence des langages acceptés aux graphes engendrés.

4.2 Ensembles rationnels de piles de piles

Dans ce paragraphe, nous introduisons la notion d'ensemble rationnel de piles de niveau k induite par le jeu d'opérations symétriques Ops_k. Intuitivement, cette notion donne une représentation finie des ensembles de piles de niveau k associés à un automate à pile sur $\mathrm{Ops}_k^\star$. Des exemples de tels ensembles sont l'ensemble de toutes les piles apparaissant dans une configuration (resp. configuration finale) accessible depuis une configuration donnée, l'ensemble des piles apparaissant dans une configuration permettant d'accéder à une configuration donnée, etc.

Rationalité au niveau 1

Au niveau 1, il est bien connu que la notion de rationalité la plus naturelle pour les piles sur un alphabet Γ est la rationalité sur le monoïde libre Γ^*. En effet, les ensembles associés aux automates à piles peuvent être décrits comme l'application d'un sous-ensemble rationnel de $\mathrm{Ops}_1^\star(\Gamma)$ à la pile vide $[\ ]_1$. Les ensembles ainsi définis sont exactement les ensembles de $\mathrm{Rat}(\Gamma^*)$. Cette propriété fondamentale des automates à pile a été établie pour la première fois par Büchi dans [Büc64]. Dans le cadre du groupe libre[5] engendré par Γ, une propriété similaire a été

5. Le monoïde $\mathrm{Ops}_1^\star(\Gamma)$ n'est pas un groupe. En effet, pop_x n'admet pas d'inverse à droite. Cependant, il vérifie les égalités $\mathrm{pop}_x\mathrm{push}_x = \mathrm{Id}_1$ pour tout $x \in \Gamma$. Cette similitude avec le groupe libre permet d'étendre la preuve du groupe libre à $\mathrm{Rat}_1(\Gamma^*)$

établie par Benois dans [Ben69] sous la forme suivante: toute partie rationnelle du groupe libre engendré par Γ qui est incluse dans Γ^* est une partie rationnelle du monoïde libre Γ^*. Cette propriété a été exploitée pour résoudre des problèmes de vérification symbolique pour les automates à pile dans [BEM97].

Nous présentons une formulation de ce résultat adaptée à notre formalisme. La preuve de cette propriété est un élégant mécanisme de saturation. Une version élaborée de ce mécanisme est à la base des résultats de normalisation présentés dans la suite de ce chapitre.

Théorème 4.2.1 ([Büc64, Ben69])**.** *Pour tout alphabet fini* Γ*,*

$$Rat(\mathrm{Ops}_1^\star(\Gamma))([\ ]_1) = Rat(\Gamma^*).$$

Démonstration. L'inclusion réciproque est immédiate. Pour l'inclusion directe, considérons un ensemble de piles dans S égal à $R([\ ]_1)$ pour un certain ensemble $R \in \mathrm{Rat}(\mathrm{Ops}_1^\star)$. Il existe un ensemble I dans $\mathrm{Rat}(\Gamma_1^*)$ tel que $R = \mathcal{R}(I)$ et donc tel que $S = \mathcal{R}(I)([\ ]_1)$.

Pour tout $\gamma \in \Gamma$, $\mathcal{R}(\gamma\bar{\gamma})$ est égal à l'élément neutre Id_k de $\mathrm{Ops}_k^\star$. L'idée est donc de calculer l'ensemble des descendants de I pour le semi-système de Thue défini par $\{(\gamma\bar{\gamma},\varepsilon) \mid \gamma \in \Gamma\}$. Nous noterons cet ensemble $I^\downarrow$. Comme le semi-système de Thue préserve l'interprétation par $\mathcal{R}$, il suit que $\mathcal{R}(I) = \mathcal{R}(I^\downarrow)$. Comme I est un ensemble rationnel, $I^\downarrow$ est aussi un ensemble rationnel. Ce résultat a été obtenu par Benois dans [Ben69]. Dans [BS86], les auteurs établissent une construction permettant d'obtenir un automate fini acceptant $I^\downarrow$ à partir d'un automate fini acceptant I travaillant en $\mathcal{O}(m^3)$ où m est le nombre d'états du premier automate.

Nous présentons un mécanisme de saturation moins efficace, mais néanmoins polynomial, qui réalise cette tâche.

Soit $A = (Q,I,F,\Delta)$ un automate fini étiqueté par Γ_1 acceptant I. Nous définissons, par récurrence, une suite d'automates $(A_i)_{i\in\mathbb{N}}$ ayant les mêmes ensembles d'états que A et tel que pour tout $i \in \mathbb{N}$, $\mathcal{R}(\mathcal{L}(A_i)) = \mathcal{R}(I)$. L'automate A_0 est égal à A. Supposons que nous ayons défini $A_i = (Q,I,F,\Delta_i)$, nous définissons $A_{i+1} = (Q,I,F,\Delta_{i+1})$ en prenant:

$$\Delta_{i+1} = \Delta_i \cup \{(p,\varepsilon,q) \mid p \xrightarrow[A_i]{\gamma\varepsilon^*\bar{\gamma}} q \text{ où } \gamma \in \Gamma\}.$$

La suite des ensembles de transitions $(\Delta_i)_{i\in\mathbb{N}}$ est croissante et bornée. Il existe donc un indice $i_0 \leq (\Gamma_1+1)\cdot|Q|^2$ tel que pour $j \geq i_0$, $A_j = A_{i_0}$. Nous noterons B l'automate obtenu après avoir éliminé les ε-transitions de A_{i_0}. Par construction, B accepte $I^\downarrow$.

Pour conclure, il suffit de remarquer que pour toute pile $s \in S$, il existe $\rho \in I^\downarrow \cap \bot_1^*\Gamma^*$ tel que $s = \mathcal{R}(\rho)([\ ]_1)$. L'ensemble de piles S est donc égal à $\pi_{\bot_1}(\mathcal{L}(B) \cap \bot_1^*\Gamma^*) \in \mathrm{Rat}(\Gamma^*)$ où $\pi_{\bot_1}$ est le morphisme alphabétique effaçant les occurrences de $\bot_1$. □

Nous définissons donc l'ensemble des langages rationnels de piles de niveau 1, noté $\mathrm{Rat}_1(\Gamma)$, comme $\mathrm{Rat}(\mathrm{Ops}_k^\star)([\ ]_1) = \mathrm{Rat}(\Gamma^*)$. Les propriétés algébriques et algorithmiques de ces ensembles sont bien connues. En particulier, un résultat fondamental est que ces ensembles forment une algèbre de Boole. Ces ensembles admettent de nombreuses caractérisations: par automates finis, par reconnaissabilité par morphisme inverse dans un monoïde fini, par expressions régulières ou par définissabilité en logique du second ordre monadique [Büc62] (pour une présentation synthétique de ces résultats voir par exemple [Wei04] et pour une présentation détaillée voir par exemple [Sak03]).

Rationalité à partir du niveau 2

Au niveau $k > 1$, nous définissons, par analogie, les ensembles rationnels[6] de piles de niveau k comme les ensembles obtenus en appliquant un sous-ensemble rationnel de $\mathrm{Ops}_k^\star$ à la pile vide de niveau k. Nous noterons $\mathrm{Rat}_k(\Gamma)$ l'ensemble de tous les ensembles rationnels de piles de niveau k.
Nous avons donc pour tout $k \geq 1$ et pour tout alphabet fini Γ,

$$\begin{aligned} \mathrm{Rat}_k(\Gamma) &= \mathrm{Rat}(\mathrm{Ops}_k^\star(\Gamma))([\ ]_k), \\ &= \mathcal{R}(\mathrm{Rat}(\Gamma_k^*))([\ ]_k). \end{aligned}$$

Remarquons que l'utilisation du jeu d'opérations symétriques Ops_k au lieu du jeu d'opérations classiques COps_k est fondamentale. Nous verrons dans le paragraphe 4.6 que si nous remplaçons $\mathrm{Ops}_k^\star$ par $\mathrm{COps}_k^\star$ dans la définition de Rat_k, nous n'obtenons plus une algèbre de Boole.

Par définition, les ensembles de $\mathrm{Rat}_k(\Gamma)$ sont naturellement liés aux automates à pile sur $\mathrm{Ops}_k(\Gamma)$ comme le montre la proposition suivante.

Proposition 4.2.2. *Pour tout automate à pile $\mathcal{A}$ sur $\mathrm{Ops}_k(\Gamma)$, l'ensemble des piles de niveau k apparaissant dans une configuration finale de $\mathcal{A}$ accessible depuis une configuration initiale est un ensemble de $Rat_k(\Gamma)$.*

Réciproquement, tout ensemble R de $Rat_k(\Gamma)$ est l'ensemble des piles apparaissant dans une configuration finale accessible depuis une configuration initiale d'un certain automate à pile A sur $\mathrm{Ops}_k(\Gamma)$.

Exemple 4.2.3. Reprenons l'automate à pile $\mathcal{A}$ sur Ops_2 de l'exemple 4.1.14. L'ensemble des piles de niveau 2 apparaissant dans une configuration finale accessible depuis la configuration initiale $(i, [\ ]_2)$ est décrit par l'ensemble R de $\mathrm{Rat}(\mathrm{Ops}_2^\star)$ défini par:

$$\{\mathrm{push}_a,\mathrm{push}_b\}^* \cdot \mathrm{copy}_1 \cdot (\mathrm{copy}_1 \cdot \{\mathrm{pop}_a,\mathrm{pop}_b\})^* \cdot T_{[\ ]_1} \cdot (\{\mathrm{push}_a,\mathrm{push}_b\} \cdot \overline{\mathrm{copy}}_1)^* \cdot \overline{\mathrm{copy}}_1$$

6. Le terme rationnel est légèrement abusif car nous ne munissons pas l'ensemble des piles de niveau k d'une structure de monoïde. Nous considérons la «projection» de $\mathrm{Rat}(\mathrm{Ops}_k^\star)$.

appliqué à la pile vide niveau 2. Cette représentation finie n'est malheureusement pas très informative et nous préférerons la représentation finie suivante

$$\{\,\mathrm{push}_a,\mathrm{push}_b\,\}^*([\,]_2).$$

Plan détaillé du chapitre

Notre but dans la suite de ce chapitre est de fournir des outils permettant de travailler avec les ensembles de $\mathrm{Rat}_k(\Gamma)$. Pour cela, nous introduisons plusieurs modèles d'automates acceptant les langages de $\mathrm{Rat}_k(\Gamma)$ qui nous permettent en particulier de dériver les propriétés de fermeture de ces langages. Nous porterons un intérêt particulier à la complexité des transformations permettant de passer d'un modèle d'accepteur fini aux autres.

Le paragraphe 4.3 présente différents accepteurs finis pour les ensembles rationnels de piles de niveau k et les transformations permettant de passer des uns aux autres. Plus précisément, nous étudierons des automates étiquetés par Γ_k et leurs versions alternantes (au sens de [CKS81]), et nous établirons l'équivalence de ces deux modèles.

Le paragraphe 4.4 présente une notion d'accepteur fini déterministe et complet pour les ensembles de $\mathrm{Rat}_k(\Gamma)$ qui permet de dériver les propriétés de clôture de ces ensembles et en particulier la fermeture par complémentaire.

Le paragraphe 4.5 étudie les relations sur les piles de niveau k induites par les ensembles rationnels de suites d'instructions de Γ_k et des instructions de tests dans $\mathrm{Rat}_k(\Gamma)$. Nous montrons en particulier que ces relations forment à chaque niveau une algèbre de Boole et nous en donnons une représentation normalisée. Cette étude étend aux niveaux supérieurs les travaux de Caucal sur les relations préfixe-reconnaissables [Cau96, Cau03a, Cau03b] qui sont les relations induites par les ensembles rationnels de suites d'instruction de Γ_1 et les instructions de tests dans $\mathrm{Rat}_1(\Gamma)$. Nous utilisons ces relations pour donner une notion d'accepteurs finis qui décrive les ensembles rationnels de Rat_k : pile de niveau $k-1$ par pile de niveau $k-1$ alors les notions d'accepteurs finis étudiés dans les paragraphes 4.3 et 4.4 «suivent» la suite réduite des piles de niveau k.

Le paragraphe 4.6 compare les notions de rationalité induites par les opérations classiques COps_k et par les opérations symétriques Ops_k. Nous établissons que les ensembles rationnels induits par COps_k sont strictement inclus dans ceux induits par Ops_k à partir du niveau 3 et qu'ils ne forment pas une algèbre de Boole.

Le paragraphe 4.7 étend la caractérisation par définissabilité en logique du second ordre monadique des ensembles rationnels de mots [Büc62]. Cette caractérisation peut être reformulée en utilisant le théorème de Rabin [Rab69] comme suit: les ensembles rationnels de mots sur $\Gamma = \{a,b\}$ sont les ensembles définis-

sables en logique du second ordre monadique sur l'arbre binaire complet étiqueté par Γ et dont les sommets sont les mots de Γ^*.
Nous montrerons que les ensembles rationnels de piles de niveau k sont les ensembles définissables en logique monadique sur le graphe $\mathrm{GStacks}_k(\Gamma)$ associé aux piles de niveau k avec les opérations de $\mathrm{Ops}_k(\Gamma)$.

Nous concluons ce chapitre par le paragraphe 4.8 où nous appliquerons les résultats obtenus à l'étude de certains enrichissements des automates finis de mots tels que les automates bidirectionnels, alternants ou automates à galets [CKS81, LLS84, GGK91, GH96] qui conservent la même expressivité que les automates fini de mots. Nous verrons que les résultats obtenus dans ce chapitre permettent de réobtenir certains résultats d'équivalence entre ces modèles et les automates finis déterministes de manière uniforme et avec une complexité minimale.

4.3 Accepteurs finis

La notion la plus naturelle d'accepteur fini pour les ensembles de Rat_k est celle d'automates finis étiquetés par les instructions de Γ_k. Pour des raisons techniques, nous allons introduire une forme légèrement normalisée qui distingue les instructions de Γ_k^{T} des instructions de Γ_k^{O}. Rappelons que pour tout ensemble P, nous désignons par $\mathrm{Sing}(P)$ l'ensemble des parties de P de cardinal au plus 1.

Définition 4.3.1. Un automate A sur Γ_k est un quadruplet (Q,I,F,Δ) où Q est un ensemble fini d'états, $I \subseteq Q$ et $F \subseteq Q$ sont respectivement les ensembles des états initiaux et finaux et $\Delta \subseteq Q \times \Gamma_k^{\mathrm{O}} \times \mathrm{Sing}(\Gamma_k^{\mathrm{T}}) \times Q$ est l'ensemble des transitions.

Une *configuration* de A est un couple (p,s) dans $Q \times \mathrm{Stacks}_k(\Gamma)$. Nous notons $\mathcal{C}_A = Q \times \mathrm{Stacks}_k(\Gamma)$ l'ensemble des configurations de A. Une transition $(p,\gamma,T,q) \in \Delta$ est notée $p \xrightarrow{\gamma} q,T$ ou simplement $p \xrightarrow{\gamma} q$ si $T = \emptyset$. Intuitivement, l'automate A peut passer de la configuration (p,s) à la configuration (q,r) en appliquant la transition $p \xrightarrow{\gamma} q,T \in \Delta$ si la pile $r = \mathcal{R}(\gamma)(s)$ est définie et si $r \in \mathrm{Dom}(\mathcal{R}(t))$ dans le cas où $T = \{\, t \,\}$.
Il convient de remarquer que l'instruction de test de pile vide t est appliquée après l'instruction γ.

L'automate A induit, pour tout $\gamma \in \Gamma_k^{\mathrm{O}}$, une relation $\xrightarrow[A]{\gamma} \subseteq \mathcal{C}_A \times \mathcal{C}_A$. Cette relation est définie, pour toutes configurations (p,s) et (q,r) de $\mathcal{C}_A$, par $(p,s) \xrightarrow[A]{\gamma} (q,r)$ s'il existe une transition $p \xrightarrow{\gamma} q,T \in \Delta$ telle $r = \mathcal{R}(\gamma)(s)$ et $r \in \mathrm{Dom}(\mathcal{R}(t))$ pour tout $t \in T$.

Un *calcul* de A est une suite $(p_0,s_0),\gamma_1,(p_1,s_1),\ldots,(p_{n-1},s_{n-1}),\gamma_n,(p_n,s_n) \in \mathcal{C}_A(\Gamma_k^{\mathrm{O}}\mathcal{C}_A)^*$ telle que pour tout $\ell \in [0,n-1]$, $(p_\ell,s_\ell) \xrightarrow[A]{\gamma_{\ell+1}} (p_{\ell+1},s_{\ell+1})$. Un calcul de

A accepte une pile $s \in \mathrm{Stacks}_k(\Gamma)$ si $p_0 \in I$, $s_0 = [\,]_k$, $p_n \in F$ et $s_n = s$. S'il existe un calcul de A acceptant une pile s, nous dirons que A accepte s et nous noterons $\mathcal{S}(A)$ l'ensemble des piles de $\mathrm{Stacks}_k(\Gamma)$ acceptées par A.

Pour l'instant, nous avons considéré les automates sur Γ_k comme des accepteurs de piles de niveau k. Ces automates peuvent être naturellement vus comme des accepteurs de suites d'instructions de Γ_k^*. L'ensemble des suites d'instructions de Γ_k acceptées par A, noté $\mathcal{I}(A)$, est l'ensemble des suites $\gamma_1 t_1 \dots t_n \gamma_n t_n \in \Gamma_k^*$ avec $n \geq 0$ telles que pour tout $i \in [1,n]$, $\gamma_i \in \Gamma_k^{\mathrm{o}}$, $t_i \in \Gamma_k^{\mathrm{T}} \cup \{\,\varepsilon\,\}$ et telles qu'il existe $\delta_1, \dots, \delta_n \in \Delta$ où pour tout $i \in [1,n]$, $\delta_i = p_{i-1} \xrightarrow{\gamma_i} p_i, T_i$ avec $p_0 \in I$, $p_{n+1} \in F$, $T_i = \{\,t_i\,\}$ si $t_i \neq \varepsilon$ et $T_i = \emptyset$ sinon.

Les deux langages acceptés par A sont liés par la relation:

$$\mathcal{S}(A) = \mathcal{R}(\mathcal{I}(A))([\,]_k).$$

Exemple 4.3.2. Nous définissons un automate $A = (Q,I,F,\Delta)$ sur Γ_2 (où $\Gamma = \{\,a\,\}$) acceptant le langage de piles $\{[\,[\,a^n\,]\,[\,a^{n-1}\,]\dots[\,a\,]\,[\,]\,]_2 \mid n > 0\}$. Les ensembles d'états de A sont donnés par $Q = \{\,i,p,q,f\,\}$, $I = \{\,i\,\}$, $F = \{\,f\,\}$. L'ensemble des transitions Δ est décrit ci-dessous:

$$i \xrightarrow{a} i \qquad i \xrightarrow{1} p \qquad p \xrightarrow{\bar{a}} q \qquad q \xrightarrow{1} p \qquad p \xrightarrow{\bar{a}} f, \perp_1.$$

Ainsi la pile $[\,[\,aa\,]\,[\,a\,]\,[\,]\,]_2$ est acceptée par le calcul:

$$\begin{aligned}&(i, [\,]_2) \xrightarrow[A]{a} (i, [\,[\,a\,]\,]_2) \xrightarrow[A]{a} (i, [\,[\,aa\,]\,]_2) \xrightarrow[A]{1} (p, [\,[\,aa\,]\,[\,aa\,]\,]_2)\\ &\xrightarrow[A]{\bar{a}} (q, [\,[\,aa\,]\,[\,a\,]\,]) \xrightarrow[A]{1} (p, [\,[\,aa\,]\,[\,a\,]\,[\,a\,]\,]_2) \xrightarrow[A]{\bar{a}} (f, [\,[\,aa\,]\,[\,a\,]\,[\,]\,]_2).\end{aligned}$$

L'ensemble $\mathcal{I}(A)$ des suites d'instructions acceptées par A est $a^+(1\bar{a})^*\perp_1$.

Exemple 4.3.3. Reprenons le langage de piles de niveau 2 présenté dans l'exemple 4.2.3. Cet ensemble est accepté par l'automate sur Γ_2 présenté dans la figure 4.1.

Remarque 4.3.4. Comme dans le cas des automates finis sur le monoïde libre, nous considérerons l'ajout d'ε-transitions aux automates sur Γ_k. Cet enrichissement peut être réalisé simplement en ajoutant un symbole ε à l'ensemble Γ_k^{o} qui s'interprète par l'application $\mathcal{R}$ comme l'identité Id_k de niveau k.

L'ajout des ε-transitions n'augmente pas l'expressivité des automates sur Γ_k. En effet, ces dernières peuvent être supprimées par un mécanisme classique de saturation [Sak03]. Cette transformation n'augmente pas le nombre d'états de l'automate et peut être réalisée en temps polynomial.
Cette notion servira uniquement à simplifier la présentation de certaines constructions.

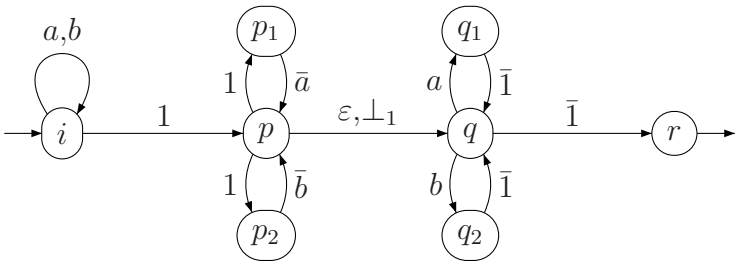


FIG. 4.1 – *Un automate sur* Γ_2 *acceptant le langage de piles de niveau 2 présenté dans l'exemple 4.2.3.*

La propriété de base des automates sur Γ_k est bien évidemment qu'ils acceptent exactement les ensembles de $\mathrm{Rat}_k(\Gamma)$.

Proposition 4.3.5. *Les automates sur* Γ_k *acceptent les ensembles de* $Rat_k(\Gamma)$.

Démonstration. Nous montrons les deux inclusions:
$\boxed{\subseteq}$ Soit A un automate sur Γ_k. Le langage de piles accepté par A est égal à $\mathcal{R}(\mathcal{I}(A))([\,]_k)$. Comme $\mathcal{I}(A)$ appartient à $\mathrm{Rat}(\Gamma_k^*)$, il suit que $\mathcal{S}(A)$ appartient à Rat_k.
$\boxed{\supseteq}$ Soit R un langage de Rat_k. Par définition de Rat_k, il existe $I \in \mathrm{Rat}(\Gamma_k^*)$ tel que $R = \mathcal{R}(I)([\,]_k)$. Remarquons que pour tout $\ell,\ell' \in [1,k]$, nous avons $\mathcal{R}(\bot_\ell \bot_{\ell'}) = \mathcal{R}(\bot_{\max\{\ell,\ell'\}})$. Il existe donc un langage $J \in \mathrm{Rat}(\Gamma_k^*)$ tel que $J \subseteq (\Gamma_k^{\mathrm{T}} \cup \{\varepsilon\}) \cdot (\Gamma_k^{\mathrm{O}} \cdot (\Gamma_k^{\mathrm{T}} \cup \{\,\varepsilon\,\}))^*$ et tel que $\mathcal{R}(I)([\,]_k) = \mathcal{R}(J)([\,]_k)$. De plus comme nous considérons $\mathcal{R}(J)$ appliqué à la pile vide de niveau k, nous pouvons dans chaque suite de J commençant par une instruction de Γ_k^{T}, supprimer cette instruction sans changer $\mathcal{R}(J)([\,]_k)$. Nous pouvons donc, sans perte de généralité, supposer que J est un sous-ensemble de $(\Gamma_k^{\mathrm{O}} \cdot (\Gamma_k^{\mathrm{T}} \cup \{\,\varepsilon\,\}))^*$.

Il est aisé de construire un automate A sur Γ acceptant le langage de suites d'instructions J. Le langage de piles $\mathcal{S}(A)$ accepté par A est donc égal à:

$$\mathcal{R}(\mathcal{I}(A))([\,]_k) = \mathcal{R}(J)([\,]_k) = \mathcal{R}(I)([\,]_k) = R.$$

□

Ces automates ne fournissent pas une représentation utile des ensembles de Rat_k comme nous l'avons vu dans les exemples 4.2.3 et 4.3.3. Nous cherchons donc à obtenir une représentation finie des ensembles de Rat_k permettant de prouver la fermeture par complémentaire.

L'idée clé est de se concentrer sur des automates qui engendrent les piles en «suivant» leurs suites réduites d'instructions. De tels automates seront appelés des automates réduits.

Définition 4.3.6. Un automate A sur Γ_k est *réduit* si pour tout calcul:

$$(p_0, [\,]_k), \gamma_1, (p_1, s_1), \gamma_2, \ldots, \gamma_n, (p_n, s_n)$$

de A, la suite $\gamma_1, \ldots, \gamma_n \in (\Gamma_k^o)^*$ est réduite.

De l'unicité de la suite réduite associée à une pile, il suit que si une pile s est acceptée par un calcul:

$$(p_0, [\,]_k) \xrightarrow[A]{\gamma_1} (p_1, s_1) \ldots (p_{n-1}, s_{n-1}) \xrightarrow[A]{\gamma_n} (p_n, s_n)$$

alors $\gamma_1 \cdot \ldots \cdot \gamma_n$ est la suite réduite de s.

Remarque 4.3.7. Une conséquence directe de cette propriété est que nous pouvons, sans perte de généralité, supposer que l'instruction $\bar{k}$ n'apparaît pas dans les transitions d'un automate réduit sur Γ_{k+1} (cf. remarque 4.1.11).

De même, nous pouvons supposer que les automates réduits sur Γ_k n'utilisent pas l'instruction $\perp_k$. En effet, nous pouvons, sans perte de généralité, supposer que les états initiaux n'apparaissent pas comme buts d'une transition de l'automate. Sous cette hypothèse, pour tout calcul de l'automate partant de la pile vide de niveau k dans un état initial et atteignant une configuration (p,s), nous avons $s = [\,]_k$ si et seulement si p est un état initial. Nous pouvons donc supprimer toutes les transitions contenant l'instruction $\perp_k$ sans changer le langage accepté.

Nous pouvons donc supposer sans perte de généralité que l'ensemble des états d'un automate réduit sur Γ_{k+1} est de la forme $Q_A \times (\Gamma_k \cup \{\, k, \varepsilon \,\})$, que l'ensemble d'états initiaux est de la forme $I_A \times \{\, \varepsilon \,\}$ et que les transitions sont de la forme $(p,\gamma) \xrightarrow{\gamma'} (q,\gamma'), T$ où $\gamma \in \Gamma_k \cup \{\, k, \varepsilon \,\}$ et $\gamma' \in \Gamma_k \cup \{\, k \,\}$ avec $\gamma' \neq \bar{\gamma}$.

Au niveau 1, les automates réduits sur Γ_1 sont simplement des automates étiquetés par Γ. Nous avons déjà vu (cf. théorème 4.2.1) qu'ils acceptent tous les ensembles de $\mathrm{Rat}_1 = \mathrm{Rat}(\Gamma^*)$.

Dès le niveau 2, cette propriété n'est plus vérifiée. Les automates réduits sur Γ_2 n'acceptent pas tous les langages de Rat_2 comme le montre l'exemple ci-dessous.

Exemple 4.3.8. Considérons par exemple le langage de piles de niveau 2

$$S = \{[\,[\,ba^n\,]\,[\,b\,]\,]_2 \mid n \geq 0\}$$

Comme S est égal à $\mathcal{R}(ba^*1\bar{a}^*\bar{b}b)([\,]_2)$, S appartient bien à Rat_2. Cependant, S n'est accepté par aucun automate réduit sur Γ_2.

Supposons par l'absurde que S soit accepté par $A = (Q_A, I_A, F_A, \Delta_A)$ réduit sur Γ_2. Considérons la pile $s = \left[\left[ba^{|Q_A|}\right][b]\right]_2$ du langage S. Sa suite réduite est $ba^{|Q_A|}1\bar{a}^{|Q_A|}$. Comme A est réduit, il existe un calcul A:

$$(p_0,s_0) \xrightarrow[A]{b} (p_1,s_1) \xrightarrow[A]{a} \cdots \xrightarrow[A]{a} (p_{|Q_A|+1},s_{|Q_A|+1}) \cdots$$
$$\cdots \xrightarrow[A]{1} (q_0,s_{|Q_A|+2}) \xrightarrow[A]{\bar{a}} \cdots \xrightarrow[A]{\bar{a}} (q_{|Q_A|},s_{2|Q_A|+2})$$

où $s_{2|Q_A|+2} = s$. Il existe $i < j \in [0,|Q_A|]$, tels que $q_i = q_j$. Il suit que la pile $\left[\left[ab^{|Q_A|}\right][ab^{j-i}]\right]_2$ est acceptée par A; ce qui apporte la contradiction avec la définition de S.

Pour rendre les automates réduits sur Γ_k suffisamment puissants pour capturer tous les langages de $\mathrm{Rat}_k(\Gamma)$, il faut les enrichir avec des opérations de *tests* (cf. sous-paragraphe 4.3.3). Une opération de test de niveau k est donnée par un langage de pile $S \subseteq \mathrm{Stacks}_\ell(\Gamma)$ pour $\ell \leq k$. Cette opération, notée Test^k_L est l'identité de niveau k restreinte à l'ensemble de piles dont la plus haute pile de niveau ℓ appartient à L. Par exemple, l'opération $\mathrm{Test}^2_{\{[b]_2\}}$ est telle que $\mathrm{Test}^2_{\{[b]_2\}}(s) = s$ si $\mathrm{top}_1(s) = [b]_1$ et n'est pas définie sinon.

Le but de ce paragraphe est de montrer que tout langage de Rat_{k+1} est accepté par un automate réduit sur Γ_{k+1} avec tests dans Rat_k (cf. sous-paragraphe 4.3.3). Ainsi, dans le cas de l'exemple 4.3.8, il suffit de rajouter le test de niveau 2 associé à l'ensemble $\{[b]_1\}$. Le langage S s'exprime alors comme $\mathcal{R}(ba^*1\bar{a}^*T^2_{\{b\}})$.

Pour caractériser ces langages de tests, il nous faut introduire un modèle d'automate plus puissant : un automate alternant sur Γ_k. Ce modèle d'automate est défini dans le sous-paragraphe 4.3.1. Intuitivement, ces automates peuvent lancer plusieurs exécutions en parallèle. Muni de cet outil, il est aisé de montrer que tout automate sur Γ_{k+1} est équivalent à un automate réduit sur Γ_{k+1} avec des tests acceptés par des automates alternants sur Γ_k.

Pour établir le résultat voulu, il nous faut montrer que les automates alternants sur Γ_k ont la même expressivité que les automates (non-alternants) sur Γ_k. Pour cela, nous passons par une forme normalisée des automates alternants qui est l'objet du sous-paragraphe 4.3.2.

Enfin dans le sous-paragraphe 4.3.2, nous établirons que les automates alternants sur Γ_k acceptent les langages de Rat_k et que donc tout automate sur Γ_{k+1} est équivalent à un automate réduit sur Γ_{k+1} avec tests dans Rat_k.

Le dernier sous-paragraphe 4.3.4 est consacré à l'étude de la complexité du test du vide des différents modèles d'automates présentés dans ce paragraphe.

Remarque 4.3.9. Nous attacherons une attention particulière à la complexité des transformations présentées. Dans les sous-paragraphes 4.3.1, 4.3.2 et 4.3.3, nous établissons de nombreuses correspondances entre les différents modèles d'au-

tomates présentés. Pour évaluer, la taille des automates, nous introduisons les ensembles de fonctions suivants: exp[0] est l'ensemble des polynômes à une variable et à coefficients polynomiaux en la taille de l'alphabet de pile $|\Gamma|$ et le niveau k et exp$[k+1]$ est l'ensemble des fonctions $n \mapsto 2^{f(n)}$ pour f dans exp$[k]$.

Pour simplifier les notations, dans une proposition qui exprime que pour tout automate A, il existe un automate B équivalent nous dirons que $|A|$ est bornée par exp$[k](|B|)$ au lieu de dire qu'il existe $f \in \exp[k]$ telle que pour tout automate A, il existe un automate B équivalent de taille inférieure à $f(|A|)$. De même nous dirons qu'un problème peut être résolu en temps exp$[k]$ au lieu de dire qu'il existe une fonction $f \in \exp[k]$ et une procédure P résolvant le problème et terminant en temps au plus $f(n)$ sur une entrée de taille n.

Enfin, dans une proposition qui exprime que pour chaque automate A, il existe un automate B acceptant le même langage, la construction de B à partir de A est toujours effective. De plus, elle peut toujours être effectuée en temps exp$[0](\max(|A|,|B|))$. La taille de B en fonction de la taille de A est donnée dans l'énoncé de la proposition.

4.3.1 Automates alternants sur Γ_k.

La notion d'alternance a été introduite dans [CKS81] et a été étendue à de nombreux modèles classiques de l'informatique théorique. Elle autorise l'exécution simultanée de plusieurs transitions d'un automate. Pour donner un sens à cette notion, dans notre cadre, il ne faut plus engendrer les piles en partant de la pile vide comme c'est le cas pour les automates sur Γ_k (cf. définition 4.3.1): il faut faire partir l'exécution de la pile à accepter. Comme le jeu d'opérations Ops_k est symétrique, ce changement de point de vue n'a pas de conséquence. Nous verrons dans le sous-paragraphe 4.6 que ce n'est pas le cas si nous considérons COps_k au lieu de Ops_k.

Définition 4.3.10. Un automate A *alternant sur* Γ_k est un uplet (Q,I,Δ) où Q est un ensemble fini d'états, $I \subseteq Q$ est l'ensemble des états initiaux et $\Delta \subseteq Q \times \mathrm{Sing}(\Gamma_k^{\mathrm{T}}) \times 2^{Q \times \Gamma_k^{\circ}}$ est l'ensemble des transitions.

Une transition $\delta = (p,t,\{(q_1,\gamma_1),\ldots,(q_n,\gamma_n)\}) \in \Delta$ sera notée $p,T \rightarrow (q_1,\gamma_1) \wedge \ldots \wedge (q_n,\gamma_n)$. Intuitivement, l'automate A dans la configuration (q,s) où $q \in Q$ et $s \in \mathrm{Stacks}_k(\Gamma)$ peut, si s satisfait le test de T (*i.e.* pour tout $t \in T$, $s \in \mathrm{Dom}(\mathcal{R}(t))$), lancer n exécutions en parallèle. La $i^{\text{ème}}$ exécution commence dans la configuration $(q_i,\mathcal{R}(\gamma_i)(s))$ (si $\mathcal{R}(\gamma_i)(s)$ est bien définie).
Nous introduisons quelques notations permettant de travailler avec ces transitions. Pour toute transition $\delta = (p,T,A) \in \Delta$, nous noterons $\mathrm{Head}(\delta) = p$, $\mathrm{Test}(\delta) = T$ et $\mathrm{Act}(\delta) = A$.

Formellement une exécution $\mathcal{E}$ de l'automate A est un couple (T,C) où T est

un arbre fini étiqueté par Γ_k° et C est une application de V_T dans l'ensemble $Q \times \text{Stacks}_k(\Gamma)$. Pour tout nœud $u \in V_T$ dont l'image par C est (p,s), il existe une transition $\delta_u = p,T \to (q_1,\gamma_1) \wedge \ldots \wedge (q_n,\gamma_n) \in \Delta$ telle que:

- pour tout $t \in T$, $s \in \text{Dom}(\mathcal{R}(t))$,
- pour tout $i \in [1,n]$, il existe $v_i \in V_T$ tel que $C(v_i) = (q_i,\mathcal{R}(\gamma_i)(s))$ et $u \xrightarrow[T]{\gamma_i} v_i$.

Nous noterons dans la suite $\Phi_\mathcal{E}$ l'application de V_T dans Δ qui associe à tout $u \in V_T$, la transition δ_u appliquée au nœud u dans l'exécution $\mathcal{E}$. Dans la suite, nous considérerons les exécutions à isomorphisme près (*i.e.* nous ne distinguerons pas deux exécutions qui ne différent que par le nommage des sommets de leurs arbres).

Nous dirons qu'une exécution $\mathcal{E} = (T,C)$ commence en $s \in \text{Stacks}_k(\Gamma)$ par l'état $q \in Q$ (resp. par la transition $\delta \in \Delta$) si $C(r(T)) = (q,s)$ (resp. $\Phi_\mathcal{E}(r(T)) = \delta$).

L'automate A accepte $s \in \text{Stacks}_k(\Gamma)$ s'il existe une exécution de A commençant en s par $i \in I$. Nous noterons $\mathcal{S}(A)$ l'ensemble des piles de $\text{Stacks}_k(\Gamma)$ acceptées par A. Par analogie nous noterons, pour tout $q \in Q$ (resp. $\delta \in \Delta$), l'ensemble $\mathcal{S}_q(A)$ (resp. $\mathcal{S}_\delta(A)$) des piles $s \in \text{Stacks}_k(\Gamma)$ telles qu'il existe une exécution de A commençant en s par l'état q (resp. par la transition δ).
Nous noterons Alt_k l'ensemble des langages de piles de niveau k acceptés par un automate alternant sur Γ_k.

Remarque 4.3.11. Remarquons les faits suivants :

1. La taille $|A|$ d'un automate alternant A sur Γ_k est borné par $|\Gamma_k|2^{2|Q_A||\Gamma_k|}$ (*i.e.*. $\exp[1](|Q_A|)$).
2. Comme pour les automates sur Γ_k, nous ajoutons des ε-transitions en rajoutant une instruction ε à Γ_k° interprétée comme Id_k. L'élimination des ε-transitions se fait par un mécanisme classique de saturation qui n'augmente pas le nombre d'états mais qui peut produire un automate de taille exponentielle en $|Q_A|$.

Exemple 4.3.12. Pour tout entier $n \geq 1$, considérons le langage rationnel L_n sur l'alphabet $\Gamma = \{0,1,\$\}$ défini par:

$$L_n = \{x\$y\$z\$y \mid y \in \{0,1\}^n \text{ et } x,z \in \Gamma^*\}$$

Considérons l'ensemble S_n des piles de niveau 2 contenant une unique pile de niveau 1 appartenant à L_n (*i.e.* $S_n = \{[\,[\,w\,]\,]_2 \mid w \in L_n\}$). Nous définissons un automate A_n alternant sur Γ_2 qui accepte S_n.

Pour tout $n \geq 1$, l'automate A_n alternant sur Γ_2 est défini par le uplet $(Q_n,\{q_0\},\Delta_n)$ où:

$$Q_n = \{q_0,q^\downarrow,f,v,t_i,p_j,r_i^x,s_i^x \mid i \in [1,n], j \in [1,n+1] \text{ et } x \in \{a,b\}\}$$

et où l'ensemble des transitions Δ_n est défini par:

$$
\begin{array}{lll}
q_0 \to (t_1,\bar{x}) \wedge (q^{\downarrow},1) & & \\
t_i \to (t_{i+1},\bar{x}) & t_n \to (v,\bar{\$}) & v \to (v,\bar{y}) \\
q^{\downarrow} \to (q^{\downarrow},\bar{y}) & q^{\downarrow} \to (p_1,\bar{\$}) & p_i \to (p_{i+1},\bar{x}) \wedge (r_i^x,y) \\
p_{n+1} \to (f,\bar{\$}) & r_i^x \to (r_i^x,y) & r_i^x \to (s_i^x,\bar{1}) \\
s_1 \to (f,\bar{x}) & s_{i+1}^x \to (s_i^x,\bar{y}) & \\
v,\{\bot_2\} \to \emptyset & f \to \emptyset &
\end{array}
$$

où $x \in \{a,b\}$, $y \in \Gamma$ et $i \in [1,n]$.

Intuitivement, l'automate lance deux exécutions. La première, utilisant les états t_i et v, vérifie que la pile est de la forme $\{[\,[\,w\$y\,]\,]_2 \mid w \in \Gamma^* \text{ et } y \in \{a,b\}^*\}$. La seconde vérifie que $\$y\$$ est bien un facteur de $w\$$. Pour effectuer cette seconde vérification, l'automate copie la pile et dépile (de manière non-déterministe) une suite de symboles se terminant par \$. Ensuite, l'automate dépile un symbole $x \in \{a,b\}$ et lance une exécution avec l'état r_1^x qui vérifie que le premier caractère en partant du haut de pile est un x. Le processus est itéré au total n fois: le dépilement du $i^{\text{ème}}$ symbole x_i lance en parallèle une exécution dans l'état r_i^x qui vérifie que le $i^{\text{ème}}$ (en partant du haut de la pile) est x_i. Cette exécution se termine en vérifiant que le $n+1^{\text{ème}}$ symbole est un \$. La figure 4.2 montre une exécution de A_2 acceptant $[\,[\,a\$ab\$ab\,]\,]_2$.

Le langage L_n est évidement rationnel et tout automate fini déterministe l'acceptant possède au moins 2^{2^n} états et donc, tout automate fini l'acceptant a au moins 2^n états. Remarquons que l'automate A_n a lui $4n+5$ états. Nous reviendrons sur cette remarque dans le paragraphe 4.8.

Une première propriété de ces automates est qu'ils acceptent tous les langages de Rat_k.
Plus précisément, en utilisant le lemme 4.1.2, nous pouvons établir l'équivalence entre les automates sur Γ_k et les automates alternants dit *élagués* dont l'arbre d'exécution est réduit à une branche et qui donc n'utilisent pas l'alternance.

Définition 4.3.13. Un automate $A = (Q_A,I_A,\Delta_A)$ est élagué si pour tout $\delta \in \Delta_A$, $|\mathrm{Act}(\delta)| \leq 1$.

La taille d'un automate alternant sur Γ_k est au plus $|Q|^2|\Gamma_k|^2$ où Q est l'ensemble des états de l'automate. Intuitivement, pour passer d'un automate sur Γ_k à un automate alternant élagué sur Γ_k, il suffit de «renverser» les transitions de l'automate et d'échanger les états initiaux et finaux. Pour la transformation réciproque, il faut en plus ajouter un nouvel état car l'exécution d'un automate alternant et élagué sur Γ_k ne se termine pas nécessairement sur la pile vide $[\,]_k$.

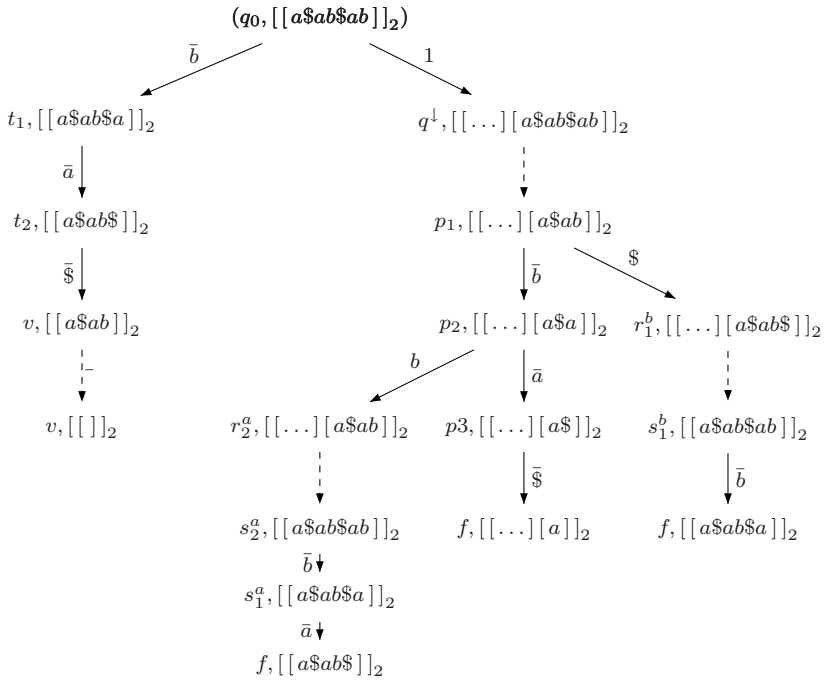


FIG. 4.2 – *Une exécution de A_2 acceptant la pile $[\,[\,a\$ab\$ab\,]\,]_2$.*

Proposition 4.3.14. *Les automates alternants élagués sur Γ_k et les automates sur Γ_k sont équivalents:*

1. *Pour tout automate A sur Γ_k, il existe un automate B alternant élagué sur Γ_k tel que $\mathcal{S}(B) = \mathcal{S}(A)$ et $|Q_B| = |Q_A|$.*
2. *Pour tout automate A alternant élagué sur Γ_k, il existe un automate B sur Γ_k tel que $\mathcal{S}(B) = \mathcal{S}(A)$ et $|Q_B| = |Q_A| + 1$.*

Démonstration. Nous établissons les deux propriétés.

1. Soit $A = (Q_A, I_A, F_A, \Delta_A)$ un automate sur Γ_k. Nous construisons un automate $B = (Q_A, F_A, \Delta_B)$ alternant élagué sur Γ_k acceptant $\mathcal{S}(A)$. L'ensemble des transitions Δ_B est défini par:

$$\begin{array}{rcl} \Delta_B & = & \{q,T \to (p,\bar{\gamma}) \mid p \xrightarrow{\gamma} q,T \in \Delta_A\} \\ & \cup & \{i,\bot_k \to \emptyset \mid i \in I_A\}. \end{array}$$

   Par le lemme 4.1.2 et par construction, $\mathcal{S}(A) = \mathcal{S}(B)$.
2. Soit $A = (Q_A, I_A, \Delta_A)$ un automate alternant élagué sur Γ_k. Nous construisons un automate $B = (Q_B, I_B, F_B, \Delta_B)$ acceptant $\mathcal{S}(A)$. L'ensemble des états $Q_B = Q_A \cup \{\bullet\}$ où $\bullet$ est un symbole n'appartenant pas à Q_A. Les états initiaux sont $I_B = \{q \in Q_A \mid q \to \emptyset \in \Delta_A\} \cup \{\bullet\}$. Les états finaux

F_B sont les états initiaux I_A de A. Enfin, l'ensemble des transitions de Δ_B est défini par:

$$\begin{array}{rcl}\Delta_B & = & \{q \xrightarrow{\bar{\gamma}} p,T \mid p,T \to q,\gamma \in \Delta_A\} \\ & \cup & \{\bullet \xrightarrow{\gamma} \bullet \mid \gamma \in \Gamma_k^\circ\} \\ & \cup & \{\bullet \xrightarrow{\gamma} p,T \mid \gamma \in \Gamma_k^\circ \text{ et } p,T \to \emptyset \in \Delta_A\}\end{array}$$

Par construction et par le lemme 4.1.2, $\mathcal{S}(A) = \mathcal{S}(B)$.

□

Exemple 4.3.15. Reprenons l'automate A sur Γ_2 présenté dans l'exemple 4.3.2. L'automate $B = (Q_B,I_B,\Delta_B)$ alternant élagué sur Γ_2 construit dans la proposition 4.3.14 est donné ci-dessous. L'ensemble des états $Q_B = \{\, i,p,q,f \,\}$, $I = \{\, f \,\}$, et l'ensemble des transitions Δ_B est:

$$\begin{array}{lll} i,\{\, \bot_2 \,\} \to \emptyset & i \to (i,\bar{a}) & p \to (i,\bar{1}) \\ q \to (p,a) & p \to (q,\bar{1}) & f,\{\, \bot_1 \,\} \to (p,a). \end{array}$$

Ainsi la pile $[\,[\,aa\,]\,[\,a\,]\,[\,]\,]_2$ est acceptée par l'exécution:

$$\begin{array}{l}(f,[\,[\,aa\,]\,[\,a\,]\,[\,]\,]_2) \xrightarrow{a} (p,[\,[\,aa\,]\,[\,a\,]\,[\,a\,]\,]_2) \xrightarrow{\bar{1}} (q,[\,[\,aa\,]\,[\,a\,]\,]_2) \\ \xrightarrow{a} (p,[\,[\,aa\,]\,[\,aa\,]\,]_2) \xrightarrow{\bar{1}} (i,[\,[\,aa\,]\,]_2) \xrightarrow{\bar{a}} (i,[\,[\,a\,]\,]_2) \xrightarrow{\bar{a}} (i,[\,]_2).\end{array}$$

Une conséquence directe de la proposition précédente est que les ensembles de Rat_k sont acceptés par les automates alternants sur Γ_k.

Une autre propriété de base des automates alternants est que les langages qu'ils acceptent sont clos par union et intersection.

Proposition 4.3.16. *Pour tout automates A et B alternant sur Γ_k, les langages $\mathcal{S}(A) \cap \mathcal{S}(B)$ et $\mathcal{S}(A) \cup \mathcal{S}(B)$ sont acceptés par des automates alternants sur Γ_k*

Démonstration. Soit $A = (Q_A,I_A,\Delta_A)$ et $B = (Q_B,I_B,\Delta_B)$ deux automates alternants sur Γ_k. Nous pouvons sans perte de généralité supposer que Q_A et Q_B sont disjoints. L'automate $C = (Q_A \cup Q_B,I_A \cup I_B,\Delta_A \cup \Delta_B)$ accepte le langage $\mathcal{S}(A) \cup \mathcal{S}(B)$. Soit $\bullet$ un symbole n'appartenant pas à $Q_A \cup Q_B$. L'automate $D = (Q_A \cup Q_B \cup \{\, \varepsilon \,\},\{\, \bullet \,\},\Delta_D)$ d'ensemble de transitions

$$\Delta_D \;=\; \Delta_A \cup \Delta_B \cup \{\bullet \to (i_A,\varepsilon) \wedge (i_B,\varepsilon) \mid i_A \in I_A \text{ et } i_b \in I_B\}.$$

accepte le langage $\mathcal{S}(A) \cap \mathcal{S}(B)$. □

Remarque 4.3.17. Nous n'obtenons pas la fermeture par complémentaire car nous n'autorisons que l'acceptation par exécution finie. Il est possible de définir une notion d'automate alternant plus générale qui autorise l'acceptation par une exécution infinie et qui implique la fermeture par complémentaire. Toutefois, cette notion n'est pas nécessaire dans le cadre de notre étude.

Nous allons maintenant présenter une propriété des exécutions des automates alternants. La proposition ci-dessous montre que nous pouvons toujours supposer que l'exécution de l'automate est telle que si à deux points de son exécution l'automate arrive dans une même configuration, alors il applique la même transition. Une telle exécution est dite *positionnelle*.

Définition 4.3.18. Une exécution $\mathcal{E} = (T,C)$ d'un automate alternant A sur Γ_k est *positionnelle* si pour tout u et $v \in V_T$, $C(u) = C(v)$ implique $\Phi_{\mathcal{E}}(u) = \Phi_{\mathcal{E}}(v)$.

Par exemple, l'exécution présentée dans la figure 4.2 est une exécution positionnelle car tous les noeuds de T sont étiquetés par des configurations différentes.

Nous allons établir que l'on peut se restreindre aux exécutions positionnelles d'un automate alternant sur Γ_k.

Proposition 4.3.19. *Pour tout automate A alternant sur Γ_k, toute pile acceptée par A est acceptée par une exécution positionnelle de A.*

Démonstration. Soient $A = (Q,I,\Delta)$ un automate alternant sur Γ_k et s une pile de $\mathrm{Stacks}_k(\Gamma)$ acceptée par A. À chaque exécution $\mathcal{E} = (T,C)$ de A, nous associons l'ensemble $X_{\mathcal{E}}$ de configurations défini par:

$$X_{\mathcal{E}} = \{c \in Q \times \mathrm{Stacks}_k(\Gamma) \mid \exists u \neq v \in V_T, C(u) = C(v) = c \text{ et } \Phi_{\mathcal{E}}(u) \neq \Phi_{\mathcal{E}}(v)\}.$$

Par définition, une exécution est positionnelle si et seulement si $X_{\mathcal{E}} = \emptyset$.

Commençons par établir que pour toute exécution $\mathcal{E}$ acceptant $s \in \mathrm{Stacks}_k(\Gamma)$ avec $X_{\mathcal{E}} \neq \emptyset$, il existe une exécution $\mathcal{E}'$ acceptant s telle que $|X_{\mathcal{E}'}| < |X_{\mathcal{E}}|$.

Soit $\mathcal{E} = (T,C)$ une exécution de A acceptant s telle que $X_{\mathcal{E}} \neq \emptyset$. Comme T est fini, il existe $c_0 \in X_{\mathcal{E}}$ et $u_0 \in V_T$ tels que $C(u_0) = c_0$ et l'image par C des autres sommets de $T_{/u_0}$ ne contient pas d'éléments dans $X_{\mathcal{E}}$ (*i.e.* $C(V_{T_{/u_0}}) \cap X_{\mathcal{E}} = \emptyset$). Soit V_0 l'ensemble des nœuds de T qui sont étiquetés par c_0 et tels qu'aucun de leurs ancêtres ne soit étiqueté par c_0:

$$V_0 = \{u \in V_T \mid C(u) = c_0 \text{ et } \forall v \neq u \in V_T, v \xrightarrow[T]{}{}^* u \Rightarrow C(v) \neq c_0\}.$$

L'exécution $\mathcal{E}'$ est construite à partir de $\mathcal{E}$ en substituant, dans $\mathcal{E}'$, le sous-arbre $T_{/u_0}$ à tous les sous-arbres enracinés en V_0. Comme par définition de V_0, les sous-arbres enracinés en V_0 sont tous disjoints, la substitution est bien définie et l'on vérifie facilement que $\mathcal{E}'$ est une exécution acceptant s. Par construction, $X'_{\mathcal{E}} \subseteq X_{\mathcal{E}}$ et par définition de $T_{/u_0}$, $c_0 \notin X'_{\mathcal{E}}$. Donc $|X'_{\mathcal{E}}| < |X_{\mathcal{E}}|$.

Soit $\mathcal{E}_0$ une exécution de A acceptant s avec $|X_{\mathcal{E}_0}|$ minimale. Il suit de ce qui précéde que $X_{\mathcal{E}_0} = \emptyset$ et donc que $\mathcal{E}_0$ est positionnelle. □

Remarque 4.3.20. La proposition 4.3.19 peut être vue comme un cas particulier du théorème 1.4.5 de détermination positionnelle des jeux de parité. En effet,

nous pouvons définir la sémantique d'un automate alternant A sur Γ_k par un jeu de parité $\mathcal{G}_A = (V_0,V_1,E = E_0 \cup E_1,\Omega)$ à deux joueurs.

$$\begin{array}{rcl} V_0 & = & Q \times \mathrm{Stacks}_k(\Gamma) \\ V_1 & = & \{(\delta,s) \mid \delta = q,T \to (q_1,\gamma_1) \wedge \ldots \wedge (q_n,\gamma_n) \\ & & \qquad \text{et } \forall j \in [1,n], s_j = \mathcal{R}(\gamma_j)(s) \text{ est défini}\} \\ E_0 & = & \{((q,s),(\delta,s)) \mid (q,s) \in V_0,(\delta,s) \in V_1 \text{ et } \mathrm{Head}(\delta) = q\} \\ E_1 & = & \{((\delta,s),(q,s')) \mid (\delta,s) \in V_1,(q,\gamma) \in \mathrm{Act}(\delta) \text{ et } s' = \mathcal{R}(s) \\ & & \qquad \text{et } \forall j \in [1,n], s_j = \mathcal{R}(\gamma_j)(s) \text{ est défini}\} \end{array}$$

La fonction Ω associe la parité 1 à tous les sommets du jeu de telle sorte que toute partie infinie soit perdante pour le Joueur 0. Il est facile de vérifier que le joueur 0 a une stratégie gagnante depuis $(q,s) \in V_0$ si et seulement s'il existe une exécution de A partant de (q,s). De plus si cette stratégie est positionnelle, l'exécution correspondante l'est aussi. La proposition 4.3.19 suit alors du théorème 1.4.5. Ce raisonnement est une adaptation immédiate de la preuve de[Wal96a, Wal02] où l'auteur montre que pour les automates d'arbres alternants avec conditions de parité il est possible de se restreindre aux exécutions positionnelles.

4.3.2 Automates alternants réduits sur Γ_k.

Dans ce sous-paragraphe, nous définissons un forme normalisée des automates alternants sur Γ_k et nous montrons que tout automate alternant est équivalent à un automate normalisé. Intuitivement, un automate normalisé (que nous appellerons *réduit*) ne peut, dans une exécution $\mathcal{E} = (T,C)$, lancer deux exécutions en parallèle avec la même instruction et ne peut visiter deux fois une même pile.

Définition 4.3.21. Un automate A alternant sur Γ_k est *réduit* si pour toute exécution $\mathcal{E} = (T,C)$ de A,

- l'arbre T est déterministe,
- pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, il existe au plus un noeud $u_s \in V_T$ tel que $C(u_s) = (q,s)$ pour un certain $q \in Q$.

La proposition ci-dessous fournit une caractérisation alternative d'un automate réduit qui permet de dériver des conditions syntaxiques assurant qu'un automate alternant sur Γ_k est réduit (cf. remarque 4.3.23).

Proposition 4.3.22. *Un automate A alternant sur Γ_k est réduit si et seulement si pour toute exécution $\mathcal{E} = (T,C)$ de A, T est déterministe et que le langage des branches de T est un ensemble de suites réduites d'instructions.*

Démonstration. Nous prouvons les deux implications.
$\boxed{\Rightarrow}$ Soit A un automate alternant sur Γ_k. Supposons, par l'absurde, qu'il existe une exécution $\mathcal{E} = (T,C)$ de A avec T déterministe et dont le langage des branches

n'est pas réduit. Il existe, donc, deux nœuds u et v de T tels que $u \xrightarrow[T]{\gamma\bar{\gamma}} v$. Si $C(u) = (p,s)$ et $C(v) = (q,s')$, nous avons par définition d'une exécution, $s' = s$ ce qui contredit le fait que A est réduit.
$\boxed{\Leftarrow}$ Soit A un automate alternant sur Γ_k tel que pour toute exécution $\mathcal{E} = (T,C)$ de A, T est déterministe et le langage des branches de T est un ensemble de suites réduites.

Supposons par l'absurde qu'il existe une exécution $\mathcal{E} = (T,C)$ avec T déterministe, une pile $s \in \text{Stacks}_k(\Gamma)$ et deux nœuds $u \neq v \in V_T$ tels que $C(u) = (q_u,s)$ et $C(v) = (q_v,s)$. Soit r l' ancêtre commun de u et de v avec $C(r) = (q_r,s')$. Il existe ρ_u et $\rho_v \in (\Gamma_k^\circ)^*$ tels que $r \xrightarrow[T]{\rho_u} u$ et $r \xrightarrow[T]{\rho_v} v$. Par définition d'une exécution, nous avons $s = \mathcal{R}(\rho_u)(s')$ et $s = \mathcal{R}(\rho_v)(s')$. Par le lemme 4.1.2, $s = R(\bar{\rho}_u\rho_v)(s)$. Comme $u \neq v$, $\bar{\rho}_u\rho_v$ n'est pas vide et par la proposition 4.1.9, $\bar{\rho}_u\rho_v$ n'est pas réduite. Comme par définition de A, ρ_u et ρ_v sont réduites, il s'en suit que $\overline{\rho_u}(|\overline{\rho_u}|) = \overline{\rho_v(1)}$. Donc $\rho_u(1) = \rho_v(1)$ ce qui contredit le fait que T est déterministe. □

Remarque 4.3.23. Remarquons les faits ci-dessous:

1. Pour tout automate $A = (Q,I,\Delta)$ alternant réduit sur Γ_k, nous pouvons, sans perte de généralité, supposer que $Q = Q' \times \Gamma_k^\circ \cup \{\bullet\}$ où $\bullet$ est un symbole n'appartenant pas à $Q' \cup \Gamma_k^\circ$, $I \subseteq Q' \times \{\bullet\}$ et que les transitions de Δ sont de la forme:

$$(p,\gamma),T \ \rightarrow ((q_1,\gamma_1),\gamma_1) \wedge \ldots \wedge ((q_n,\gamma_n),\gamma_n)$$

   avec pour tout $i \neq j \in [1,n]$, $\gamma_i \neq \gamma_j$ et pour tout $i \in [1,n]$, $\gamma_i \neq \bar{\gamma}$ si $\gamma \neq \bullet$. Dans la suite, nous supposerons toujours que les automates alternants réduits considérés sont de cette forme.
2. Toutes les exécutions d'un automate réduit sont positionnelles.
3. La taille d'un automate alternant réduit sur Γ_k est bornée par $\Gamma_k|Q|^{\Gamma_k+1}$. Si l'on fixe Γ, $|A|$ est donc polynomial en son nombre d'états.

En adaptant légèrement les constructions de proposition 4.3.14, nous pouvons montrer que, pour tout automate alternant élagué et réduit sur Γ_k, il existe un automate réduit sur Γ_k équivalent (et vice-versa).

Proposition 4.3.24. *Les automates alternants élagués réduits sur Γ_k et les automates réduits sur Γ_k sont équivalents:*

1. *Pour tout automate A alternant élagué réduit sur Γ_k, il existe un automate B sur Γ_k tel que $\mathcal{S}(B) = \mathcal{S}(A)$ et $|Q_B| = |Q_A|$.*
2. *Pour tout automate A réduit sur Γ_k, il existe un automate B alternant élagué réduit sur Γ_k tel que $\mathcal{S}(B) = \mathcal{S}(A)$ et $|Q_B| \leq |\Gamma_k| \cdot (|Q_A| + 1)$.*

Démonstration. Nous établissons les deux propositions.

1. Nous adaptons la construction 2 de la preuve de la proposition 4.3.14 de manière à garantir que l'automate sur Γ_k soit réduit. Soit $A = (Q_A,I_A,\Delta_A)$ un automate alternant, élagué et réduit.
   Nous construisons un automate $B = (Q_B,I_B,F_B,\mu_B,\Delta_B)$ réduit sur Γ_k équivalent à A.
   L'ensemble des états Q_B est $(Q_A \cup \{\bullet\}) \times (\Gamma_k^o \cup \{\varepsilon\})$ où $\bullet$ est un symbole n'appartenant pas à Q_A. Les états initiaux sont $I_B = \{(q,\varepsilon) \mid q,T \to \emptyset \in \Delta_A\} \cup \{(\bullet,\varepsilon)\}$. Les état finaux sont $F_B = I_A \times (\Gamma_k^o \cup \{\varepsilon\})$. Enfin, l'ensemble des transitions de Δ_B est défini par:

$$\begin{aligned}\Delta_B &= \{(q,\gamma) \xrightarrow{\gamma'} (p,\gamma'),T \mid p,T \to (q,\bar{\gamma}') \in \Delta_A \text{ et } \gamma' \neq \bar{\gamma} \neq \varepsilon\} \\ &\cup \{(\bullet,\gamma) \xrightarrow{\gamma'} (p,\gamma'),T \mid p,T \to \emptyset \in \Delta_A \text{ et } \gamma' \neq \bar{\gamma} \neq \varepsilon\} \\ &\cup \{(\bullet,\gamma) \xrightarrow{\gamma'} (\bullet,\gamma') \mid \gamma' \neq \bar{\gamma} \neq \varepsilon\}\end{aligned}$$

   où γ et γ' appartiennent à $\Gamma_k^o \cup \{\varepsilon\}$.
   Par construction et par le lemme 4.1.2, $\mathcal{S}(A) = \mathcal{S}(B)$.
2. Il suffit de remarquer que la construction 1 de la preuve de la proposition 4.3.14 produit un automate alternant et élagué réduit si l'automate sur Γ_k fourni en entrée est réduit.

□

4.3.2.1 Équivalence entre automates alternants et automates alternants réduits.

Nous établissons que les automates alternants sur Γ_k sont équivalents aux automates alternants réduits sur Γ_k. Comme les automates alternants et réduits sont, en particulier, des automates alternants, il suffit de construire pour tout automate A alternant sur Γ_k, un automate B alternant réduit équivalent. Intuitivement, une exécution acceptante de B va simuler une exécution acceptante positionnelle de A.

Proposition 4.3.25. *Pour tout automate $A = (Q_A,I_A,\Delta_A)$ alternant sur Γ_k, il existe un automate $B = (Q_B,I_B,\Delta_B)$ alternant réduit sur Γ_k tel que $\mathcal{S}(A) = \mathcal{S}(B)$. De plus, $|Q_B|$ est bornée par* $\exp[1](|Q_A|)$.

Démonstration. Soit $A = (Q_A,I_A,\Delta_A)$ un automate alternant sur Γ_k. Nous commençons par éliminer les ε-transitions de A. Cette transformation n'augmente pas le nombre d'états de A et peut être réalisée en temps $\exp[1](|Q_A|)$.

Nous construisons un automate $B = (Q_B,I_B,\Delta_B)$ alternant réduit sur Γ_k équivalent à A. L'automate B est construit de manière à obtenir une bijection entre les exécutions de B et les exécutions positionnelles de A.

L'ensemble des états de B est:

$$Q_B = (\Gamma_k^o \cup \{\bullet\}) \times \mathrm{Strat}_A \times 2^{Q_A \times Q_A} \times 2^{Q_A \times Q_A}$$

où Strat_A est l'ensemble des fonctions partielles f de Q_A dans Δ_A telles que pour tout $q \in \mathrm{Dom}(f)$, $\mathrm{Head}(f(q)) = q$. Pour toute fonction $f \in \mathrm{Strat}_A$ et pour tout $\gamma \in \Gamma_k^o$, nous définissons la relation $\xrightarrow[f]{\gamma}$ par:

$$\xrightarrow[f]{\gamma} = \{(p,q) \mid p \in \mathrm{Dom}(f) \text{ et } (q,\gamma) \in \mathrm{Act}(f(p))\}.$$

Les transitions de B contiennent deux types de transitions. Une transition de la forme:

$$(\gamma_0,f,R^{\uparrow},R^{\downarrow}),T \rightarrow \bigwedge_{\gamma \in R} \left((\gamma,f_\gamma,R_\gamma^{\uparrow},R_\gamma^{\downarrow}),\gamma\right)$$

avec $\gamma_0 \in \Gamma_k^o$, $R \subseteq \Gamma_k^o \backslash \{\bar{\gamma}_0\}$, $f,f_\gamma \in \mathrm{Strat}_A$ et $R^{\uparrow},R_\gamma^{\uparrow},R^{\downarrow},R_\gamma^{\downarrow} \subseteq Q_A \times Q_A$ appartient à Δ_B si:

1. pour tout $\gamma \in \Gamma_k^o \setminus \{\bar{\gamma}_0\}$ et pour tout $p,q \in Q_A$:

$$\begin{cases} p \xrightarrow[f]{\gamma} q & \Rightarrow \quad \gamma \in R \text{ et } q \in \mathrm{Dom}(f_\gamma) \\ p \xrightarrow[f_\gamma]{\bar{\gamma}} q & \Rightarrow \quad q \in \mathrm{Dom}(f) \end{cases}$$

2. $T = \max\{t \in Test(\delta) \mid \delta \in \mathrm{Im}(f)\}$
3. $$R^{\downarrow} = \left(\bigcup_{\gamma \in R} (\xrightarrow[f]{\gamma} \cdot R_\gamma^{\downarrow} \cdot \xrightarrow[f_\gamma]{\bar{\gamma}} \cup \xrightarrow[f]{\gamma} \cdot \xrightarrow[f_\gamma]{\bar{\gamma}}) \right)^+$$
4. pour tout $\gamma \in R$,

$$R_\gamma^{\uparrow} = \xrightarrow[f_\gamma]{\bar{\gamma}} \cdot \left(\left(\bigcup_{\gamma' \in R \setminus \{\gamma\}} \xrightarrow[f]{\gamma'} \cdot R_{\gamma'}^{\downarrow} \cdot \xrightarrow[f_{\gamma'}]{\bar{\gamma}'} \cup \xrightarrow[f]{\gamma'} \cdot \xrightarrow[f_{\gamma'}]{\bar{\gamma}'} \right) \cup R^{\uparrow} \right)^* \cdot \xrightarrow[f]{\gamma}$$

5. $$(R^{\uparrow} \cup R^{\downarrow})^+ \cap \{(q,q) \mid q \in Q_A\} = \emptyset.$$
6. pour tout $\gamma \in R$,

$$\mathrm{Dom}(f_\gamma) = \left(\xrightarrow[f]{\gamma} \cdot (R_\gamma^{\downarrow})^* \right) (\mathrm{Dom}(f)).$$

Le deuxième type de transitions correspond aux «transitions initiales». Une transition de la forme:

$$(\bullet,f,\emptyset,R^{\downarrow}),T \to \bigwedge_{\gamma\in R} \left(\left(\gamma,f_\gamma,R^{\uparrow}_\gamma,R^{\downarrow}_\gamma\right),\gamma\right)$$

où $R \subseteq \Gamma^{\circ}_k$, $f,f_\gamma \in \mathrm{Strat}_A$ et $R^{\uparrow}_\gamma,R^{\downarrow},R^{\downarrow}_\gamma \subseteq Q_A \times Q_A$ appartient à Δ_B si elle satisfait les conditions 1,2,3,4 et 5 ainsi que la condition 7 présentée ci-dessous:

7. il existe $i_0 \in I \cap \mathrm{Dom}(f)$ tel que pour tout $p \in Q_A$,

$$p \in \mathrm{Dom}(f) \Leftrightarrow (i_0,p) \in R^{\downarrow}.$$

L'ensemble des états initiaux I_B est égal à $\{(\bullet,f,\emptyset,R^{\downarrow}) \mid f \in \mathrm{Strat}_A, R^{\downarrow} \subseteq Q_A \times Q_A\}$.

Intuitivement, si dans une exécution $\mathcal{E}_B = (T_B,C_B)$, un nœud u de T_B est étiqueté par $C_B(u) = ((\gamma,f,R^{\uparrow},R^{\downarrow}),s)$ alors il existe une exécution positionnelle de A qui quand elle visite la pile s adopte la «stratégie» décrite par f. Les ensembles $R^{\uparrow}$ et $R^{\downarrow}$ servent à garantir que l'exécution positionnelle de A suivant la stratégie décrite par $\mathcal{E}_B$ est bien finie (cf. exemple 4.3.26).

Nous définissons maintenant une application Ψ entre les exécutions positionnelles acceptantes de A et les exécutions acceptantes de B.

Définition de Ψ.

Pour toute exécution positionnelle $\mathcal{E}_A = (T_A,C_A)$ de A commençant par (i_0,s_0), l'exécution $\Psi(\mathcal{E}_A) = \mathcal{E}_B = (T_B,C_B)$ est définie comme suit. L'arbre T_B est défini par:

$$\begin{array}{lcl} V_{T_B} & = & \pi_2(C_A(V_{T_A})) \\ T_B & = & \{(s,\gamma,s') \in V_{T_B} \times \Gamma^{\circ}_k \times V_{T_B} \mid \rho_{s_0,s'} = \rho_{s_0,s}\gamma\}. \end{array}$$

où $\rho_{s_0,s}$ (resp. $\rho_{s_0,s'}$) désigne la suite réduite transformant s_0 en s (resp. s').

Par la proposition 4.1.9, il suit que T_B est déterministe. Comme T_A est fini, T_B l'est aussi.

Nous définissons maintenant l'application C_B. Pour tout $s \in V_{T_B}$, nous définissons $C_B(s) = ((\gamma,f,R^{\uparrow},R^{\downarrow}),s)$ par:

- $\gamma = \bullet$ si $s = s_0$ et sinon $\gamma \in \Gamma^{\circ}_k$ est tel qu'il existe $s' \in V_{T_B}$ avec $s' \xrightarrow[T_B]{\gamma} s$,
- la fonction partielle f est la fonction de Strat_A définie pour tout $q \in Q_A$ par:

$$f(q) = \begin{cases} \Phi_{\mathcal{E}_A}(v) & \text{s'il existe } v \in V_{T_A},\ C_A(v) = (q,s) \\ \text{non définie} & \text{sinon} \end{cases}$$

Comme $\mathcal{E}_A$ est positionnelle, f appartient bien à Strat_A.

- si $\gamma = \bullet$, alors $R^{\uparrow} = \emptyset$ et sinon $R^{\uparrow} \subseteq Q_A \times Q_A$ est l'ensemble des couples $(p,q) \in Q_A \times Q_A$ tels qu'il existe une suite $u \xrightarrow[T_A]{\bar{\gamma}} u_0 \xrightarrow[T_A]{\gamma_1} \ldots \xrightarrow[T_A]{\gamma_n} u_n \xrightarrow[T_A]{\gamma} v$ avec $n > 0$ telle que $C_A(u) = (p,s)$, $C_A(v) = (q,s)$ et $s \notin \pi_2(\{C_A(u_i) \mid i \in [0,n]\})$.
- $R^{\downarrow}$ est l'ensemble des couples d'états $(p,q) \in Q_A \times Q_A$ tels qu'il existe un calcul $u_0 \xrightarrow[T_A]{\gamma_1} \ldots \xrightarrow[T_A]{\gamma_n} u_n$ avec $n > 0$ tel que $C_A(u_0) = (p,s)$, $C_A(u_n) = (q,s)$ et $\mathcal{R}(\bar{\gamma})(s)$ n'appartient pas à $\pi_2(\{C_A(u_i) \mid i \in [0,n]\})$ (si $\gamma \neq \bullet$).

Il nous faut vérifier que $\mathcal{E}_B$ est bien une exécution de B. Pour tout $s \in V_{T_B}$, nous supposerons que $C(s) = (q_s,s)$ et $q_s = (\gamma_s,f_s,R_s^{\uparrow},R_s^{\downarrow})$. Il nous faut montrer que pour tout $s \in V_{T_B}$, la transition δ_s définie ci-dessous appartient à Δ_B.

$$\delta_s = q_s,\{t\} \rightarrow \bigwedge_{\gamma \in R} (q_{s_\gamma},\gamma).$$

où $t = \max\{t \in Test(\delta) \mid \delta \in \mathrm{Im}(f_s)\}$, $R = \{\gamma \in \Gamma_k^{\mathrm{o}} \mid s \xrightarrow[T_B]{\gamma} s_\gamma\}$ et $s_\gamma = \mathcal{R}(\gamma)(s)$. Par définition de T_B, $\bar{\gamma}_s$ n'appartient pas à R.

Nous distinguons deux cas selon que s est égale à s_0 ou non.

Cas $s \neq s_0$. Il faut établir que δ_s vérifie les conditions 1,2,3,4,5 et 6 de la définition de Δ_B.

1. Soit $\gamma \in \Gamma_k^{\mathrm{o}} \setminus \{\bar{\gamma}_s\}$ et soient $p,q \in Q$ tels que $p \xrightarrow[f_s]{\gamma} q$. Par définition de $\mathcal{E}_B$, il existe $u \in V_{T_A}$ tels que $C_A(u) = (p,s)$ et $(q,\gamma) \in \mathrm{Act}(\Phi_A(u))$. Donc il existe $v \in V_{T_A}$ tel que $C_A(v) = (q,\mathcal{R}(\gamma)(s))$. Donc $q \in \mathrm{Dom}(f_{s_\gamma})$. L'autre implication est similaire.
2. Immédiate par définition de δ_s.
3. Il nous faut établir que:

$$R_s^{\downarrow} = \left(\bigcup_{s \xrightarrow[T_B]{\gamma} s_\gamma} (\xrightarrow[f]{\gamma} \cdot R_{s_\gamma}^{\downarrow} \cdot \xrightarrow[f_\gamma]{\bar{\gamma}} \cup \xrightarrow[f]{\gamma} \cdot \xrightarrow[f_\gamma]{\bar{\gamma}}) \right)^{+}$$

   L'inclusion réciproque suit de la définition de $R_s^{\downarrow}$ et des $R_{s_\gamma}^{\downarrow}$. L'inclusion directe est une conséquence immédiate du lemme 4.1.12.
4. Cette condition est similaire à la précédente.
5. Supposons par l'absurde qu'il existe $p \in Q_A$ tel que $(p,p) \in (R_s^{\uparrow} \cup R_s^{\downarrow})^{+}$. Il suit de la définition de ces ensembles qu'il existe $u,v \in V_{T_A}$ tels que $u \xrightarrow[T_A]{}{}^{*} v$ et $C_A(u) = (p,s)$ et $C_B(v) = (p,s)$. Comme $\mathcal{E}_A$ est positionnelle, il suit que T_A contient une branche infinie; ce qui contredit le fait que T_A est fini.

6. Soit $\gamma \in R$, montrons que $\mathrm{Dom}(f_{s_\gamma}) = \xrightarrow[f_s]{\gamma} \cdot R^{\downarrow}_{s_\gamma}(\mathrm{Dom}(f_s))$. L'inclusion réciproque suit de la condition 1 et de la définition de $R^{\downarrow}_{s_\gamma}$. Pour l'inclusion directe, supposons, par l'absurde, qu'il existe $p \in \mathrm{Dom}(f_{s_\gamma})$ et $p \notin \xrightarrow[f_s]{\gamma} \cdot R^{\downarrow}_{s_\gamma}(\mathrm{Dom}(f_s))$. Comme $p \in \mathrm{Dom}(f_{s_\gamma})$, il existe un chemin $u_0 \xrightarrow[T_A]{\gamma_1} u_1 \ldots u_{n-1} \xrightarrow[T_A]{\gamma_n} u_n$ tel que $u_0 = r(T_A)$ et $C(u_n) = (p,s_\gamma)$. Comme par définition de T_B $\rho_{s_0,s_\gamma} = \rho_{s_0,s}\gamma$, il suit qu'il existe $\ell \in [1,n-1]$ tel que $C(u_\ell) = (p_\ell,s)$. Soit ℓ_0 le plus grand indice satisfaisant cette propriété. Par définition de f_s, p_{ℓ_0} appartient à $\mathrm{Dom}(f_s)$. Par maximalité de ℓ_0, $(p_{\ell_0},p) \in \mathcal{R}^{\downarrow}_{s_\gamma}$ et donc $p \in \xrightarrow[f_s]{\gamma} \cdot R^{\downarrow}_{s_\gamma}(\mathrm{Dom}(f_s))$; ce qui amène la contradiction.

Ψ est une injection. Soient $\mathcal{E}_A = (T_A,C_A)$ et $\mathcal{E}'_A = (T'_A,C'_A)$ deux exécutions positionnelles de A distinctes de A. Montrons que $\Psi(\mathcal{E}_A) \neq \Psi(\mathcal{E}'_A)$.
Si $C_A(r(T_A)) = C'_A(r(T'_A))$ alors comme $\mathcal{E}_A \neq \mathcal{E}'_A$, les stratégies associées aux deux exécutions différent et donc $\Psi(\mathcal{E}_A) \neq \Psi(\mathcal{E}'_A)$.
Si $\pi_2(C_A(r(T_A))) = \pi_2(C'_A(r(T'_A)))$ et $i_A = \pi_1(C_A(r(T_A))) \neq \pi_1(C'_A(r(T'_A))) = i_{A'}$, remarquons qu'à cause de la condition 5, l'état i satisfaisant la condition 7 est unique. Par la condition 7, les images par Ψ de $\mathcal{E}_A$ et $\mathcal{E}'_A$ sont différentes car dans un cas i_A remplit la condition 7 et dans l'autre c'est i'_A.
Si $\pi_2(C_A(r(T_A))) \neq \pi_2(C'_A(r(T'_A)))$, par définition de Ψ, $\Psi(\mathcal{E}_A) \neq \Psi(\mathcal{E}'_A)$.

Dans tous les cas, nous avons $\Psi(\mathcal{E}_A) \neq \Psi(\mathcal{E}_B)$.

Ψ est une surjection. Soit $\mathcal{E}_B = (T_B,C_B)$ une exécution acceptante de B acceptant la pile s_0. Nous allons construire une exécution positionnelle $\mathcal{E}_A = (T_A,C_A)$ de A telle que $\Psi(\mathcal{E}_A) = \mathcal{E}_B$.

Intuitivement, $\mathcal{E}_A$ est l'exécution positionnelle de A qui suit la stratégie définie par l'exécution $\mathcal{E}_B$. Comme B est réduit, nous pouvons sans perte de généralité supposer que $V_{T_B} \subseteq \mathrm{Stacks}_k(\Gamma)$ et que pour tout $s \in V_{T_B}$, $C_B(s) = (q_s,s)$. De plus, nous supposerons dans la suite $q_s = (\gamma_s,f_s,R^{\uparrow}_s,R^{\downarrow}_s)$. Par définition de I_B et de Δ_B, il existe un unique $i_0 \in \mathrm{Dom}(f_{s_0}) \cap I_A$ satisfaisant la condition 7.

Nous définissons $\mathcal{E}_A = (T_A,C_A)$ comme suit :

$$\begin{aligned} V_{T_A} &= \{q_0,s_0,\gamma_1,q_1,s_1,\ldots,\gamma_n,q_n,s_n \mid q_0 = i_0, \\ &\qquad \forall \ell \in [0,n-1], q_i \xrightarrow[f_{s_i}]{\gamma_{i+1}} q_{i+1} \text{ et } s_{i+1} = \mathcal{R}(\gamma_{i+1})(s_i)\}, \\ T_A &= \{(u,\gamma,u\gamma qs) \in V_{T_A} \times \Gamma^{\circ}_k \times V_{T_A} \mid q \in Q_A \text{ et } s \in \mathrm{Stacks}_k(\Gamma)\}. \end{aligned}$$

et pour tout $uqs \in V_{T_A}$ (où $q \in Q_A$ et $s \in \mathrm{Stacks}_k(\Gamma)$),

$$C_A(uqs) = (q,s).$$

Par construction T_A est un arbre déterministe. Pour établir que $\mathcal{E}_A$ est une exécution, il suffit de montrer que T_A est fini. Comme le montre l' exemple 4.3.26, la finitude de T_A est garantie par les ensembles $R^{\uparrow}$ et $R^{\downarrow}$.

Exemple 4.3.26. Considérons l'automate $C = (Q_C,I_C,\Delta_C)$ avec $Q_C = \{i,p\}$, $I_C = \{i\}$ et $\Delta_C = \{i \to (p,a),p \to (i,\bar{a})\}$. Le langage $\mathcal{S}(C)$ est vide car une exécution acceptante de C ne peut être finie.

Si nous omettions les ensembles $R^{\uparrow}$ et $R^{\downarrow}$ de la construction de l'automate réduit, nous obtiendrions un automate D avec deux transitions:

$$(\bullet,f_0) \to ((a,f_1),a) \qquad (a,f_1) \to \emptyset$$

où f_0 et f_1 sont définies par $f_0(i) = \delta_0$ et $f_1(p) = \delta_1$. Le langage accepté par D est non vide.

Si maintenant nous ajoutons les ensembles $R^{\uparrow}$ et $R^{\downarrow}$, le langage accepté est vide. En effet, l'automate réduit construit dans cette preuve ne contient pas de «transition initiale». En effet, si une transition $\delta = (\bullet,f,R^{\uparrow},R^{\downarrow}) \to A$ appartient à l'ensemble des transitions, l'état initial i appartient à $\mathrm{Dom}(f)$ par la condition 7. Par la condition 3, il suit que $R^{\downarrow}$ contient (i,i). Donc δ ne peut satisfaire la condition 5.

Avant d'établir que T_A est fini (cf. fait 4), il nous faut établir des propriétés liant les ensembles $R^{\uparrow}$ et $R^{\downarrow}$ apparaissant dans $\mathrm{Im}(C_B)$ et l'exécution $\mathcal{E}_A$.

Fait 1. *Par construction de T_A, nous avons $\pi_2(C_A(V_{T_A})) \subseteq V_{T_B}$ et pour tout $u \in V_{T_A}$, $C_A(u) = (p,s)$ implique $p \in Dom(f_s)$.*

Fait 2. *Pour tout $u,v \in V_{T_A}$ tels que $C_A(u) = (p,s)$, $C_A(v) = (q,s)$ et $u \xrightarrow[T_A]{\rho} v$ avec pour tout $\rho' \sqsubset \rho$, $\mathcal{R}(\rho')(s) \neq \mathcal{R}(\rho)(s)$, $(p,q) \in R^{\downarrow}_s$.*

La preuve procède par récurrence sur la longueur de ρ en utilisant le lemme 4.1.12 et la condition 3.

Fait 3. *Pour tout $s \in V_{T_B}$ tel que $\gamma_s \neq \bullet$ et pour tous $u,v \in V_{T_A}$ tels que $C_A(u) = (p,s)$, $C_A(v) = (q,s)$ et il existe $u \xrightarrow[T_A]{\bar{\gamma}_s} u_0 \xrightarrow[T_A]{\gamma_1} \dots \xrightarrow[T_A]{\gamma_n} u_n \xrightarrow[T_A]{\gamma_s} v$ tel que $s \notin \pi_2(\{C_A(u_i) \mid i \in [0,n]\})$, $(p,q) \in R^{\uparrow}_s$.*

La preuve procède par récurrence sur la hauteur de s dans T_B en utilisant le lemme 4.1.12, le fait 2 et la condition 4.

Fait 4. *Nous pouvons maintenant établir que T_A est fini.*

Supposons, par l'absurde, que T_A est infini. Comme T_A est déterministe, par le lemme de König, T_A a une branche infinie. D'après le fait 1, l'ensemble $C_A(V_{T_A})$ est fini et il existe donc u et $v \in V_{T_A}$ tels que $C_A(u) = C_A(v) = (p,s)$ et $u \xrightarrow[T_A]{\rho} v$.

Par le lemme 4.1.12 et par les faits 2 et 3, il suit que $(p,p) \in (R_s^{\downarrow} \cup R_s^{\uparrow})^+$ ce qui contredit la condition 6.

Nous dérivons la réciproque du fait 2.

Fait 5. *Pour tout* $s \in \mathrm{Stacks}_k(\Gamma)$ *et pour tout* $(p,q) \in R^{\downarrow}$ *tel que* $(p,s) \in C_A(V_{T_A})$, *il existe* $u,v \in V_{T_A}$ *tel que* $C_A(u) = (p,s)$ *et* $C_A(v) = (q,s)$, $u \xrightarrow[T_A]{\rho} v$ *avec pour tout* $\rho' \sqsubset \rho$, $\mathcal{R}(\rho')(s) \neq \mathcal{R}(\bar{\gamma})(s)$.
La preuve procède par récurrence sur la structure de l'arbre T_A (qui est fini) en utilisant la condition 3 et le lemme 4.1.12.

Nous pouvons maintenant établir la réciproque du fait 1.

Fait 6. *Pour tout* $s \in V_{T_B}$ *et pour tout* $p \in Dom(f_s)$, *il existe* $u \in V_{T_A}$ *tel que* $C_A(u) = (p,s)$.

La preuve procède par récurrence sur la hauteur de s dans T_B en utilisant le fait 4 et la condition 6 et 7.

Enfin, nous pouvons établir la réciproque du fait 3.

Fait 7. *Pour tout* $s \in V_{T_B}$ *tel que* $\gamma_s \neq \bullet$ *et* $(p,q) \in R_s^{\uparrow}$, *il existe une suite* $u \xrightarrow[T_A]{\bar{\gamma}} u_0 \xrightarrow[T_A]{\gamma_1} \ldots \xrightarrow[T_A]{\gamma_n} u_n \xrightarrow[T_A]{\gamma} v$ *avec* $n > 0$ *telle que* $C_A(u) = (p,s)$, $C_A(v) = (q,s)$ *et* $s \notin \pi_2(\{C_A(u_i) \mid i \in [0,n]\})$. La preuve procède par récurrence sur la hauteur de s dans T_B en utilisant les faits 4 et 5 et la condition 4.

Par le fait 4 et par construction de $\mathcal{E}_A$, $\mathcal{E}_A$ est une exécution acceptante de A. Et par les faits 1 à 7, $\mathcal{E}_B$ est l'image par Ψ de $\mathcal{E}_A$.

Conclusion. Nous avons donc établi que Ψ est une bijection des exécutions positionnelles acceptantes de A dans les exécutions acceptantes de B. Nous avons donc:

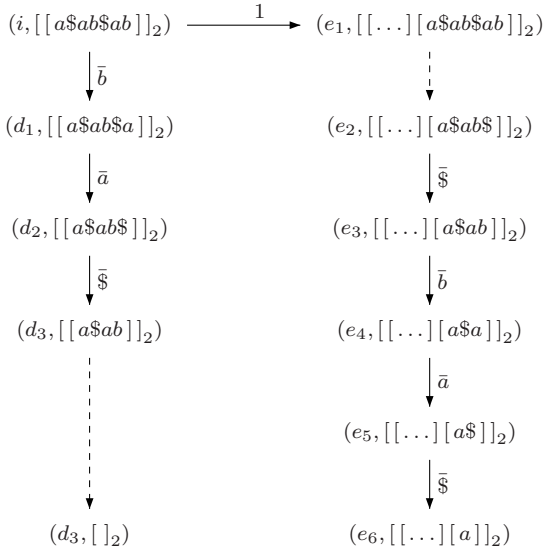
$$\mathcal{S}(A) = \mathcal{S}(B).$$

□

Exemple 4.3.27. Considérons l'automate A_2 alternant sur Γ_2 présenté dans l'exemple 4.3.12. La preuve de la proposition 4.3.25 construit un automate B_2 alternant réduit sur Γ_2 acceptant le même langage que A_2. La figure 4.3 présente l'exécution de B_2 correspondant à l'exécution de A_2 présentée dans la figure 4.2.

4.3.2.2 Équivalence alternants et non-alternants au niveau 1.

Nous concluons ce paragraphe en donnant une première application de l'équivalence entre les automates alternants sur Γ_k et les automates alternants et réduits sur Γ_k. Nous allons montrer qu'au niveau 1, les automates alternants sur Γ_1 acceptent exactement les ensembles de Rat_1.



$$
\begin{array}{lcl}
i & = & (\bullet,f_i,\emptyset,\{\,(q_0,s_1^b),(q_0,s_2^a)\,\}) \\
e_1 & = & (1,f_{e_1},\emptyset,\{\,(q^{\downarrow},r_1^b),(q^{\downarrow},r_2^a)\,\}) \\
e_2 & = & (\bar{a},f_{e_2},\emptyset,\{\,(q^{\downarrow},r_1^b),(q^{\downarrow},r_2^a)\,\}) \\
e_3 & = & (\$,f_{e_3},\emptyset,\{\,(p_1,r_2^a)\,\}) \\
e_4 & = & (\bar{b},f_{e_4},\emptyset,\emptyset) \\
e_5 & = & (\bar{a},f_{e_5},\emptyset,\emptyset) \\
e_6 & = & (\bar{\$},f_{e_6},\emptyset,\emptyset) \\
d_1 & = & (\bar{\$},f_{d_1},\emptyset,\emptyset) \\
d_2 & = & (\bar{b},f_{d_2},\emptyset,\emptyset) \\
d_3 & = & (\bar{a},f_{d_3},\emptyset,\emptyset) \\
d_4 & = & (\bar{a},f_{d_4},\emptyset,\emptyset)
\end{array}
$$

FIG. 4.3 – *L'exécution de l'automate B_2 acceptant $[\,[a\$ab\$ab]\,]_2$.*

Au niveau 1, cette propriété peut être obtenue en remarquant que l'ensemble des piles acceptées par un automate alternant sur Γ_1 correspond à un ensemble de sommets de l'arbre binaire complet Δ_2 définissable en logique du second ordre monadique. Il suit alors du théorème de Rabin [Rab69] que cet ensemble est rationnel.

Nous présentons une preuve reposant sur la proposition 4.3.25. La complexité de la transformation est moins élevée qu'en passant par la définissabilité en logique du second ordre monadique. De plus, cette preuve se généralise à tout niveau (cf. proposition 4.3.41).

Nous savons par la propriété 4.3.25 que nous pouvons nous concentrer sur les automates alternants réduits sur Γ_1. Nous «élaguons» ces automates. Nous donnons l'intuition de cette transformation sur un exemple simple d'automate alternant réduit sur Γ_1.

Exemple 4.3.28. Considérons un automate $A = (Q_A,I_A,\Delta_A)$ alternant réduit sur Γ_1 où $Q_A = \{\,i,p,q,r\,\}$, $I_A = \{\,i\,\}$ et où Δ_A est l'ensemble des transitions:

$$
\begin{array}{lll}
\delta_1 = i \to (i,\bar{b}) \wedge (p,a) & \delta_2 = i \to (i,\bar{a}) \wedge (q,b) & \delta_3 = i,\bot_1 \to \emptyset \\
\delta_4 = p \to (p,a) \wedge (p,b) & \delta_5 = p \to \emptyset & \delta_6 = q \to (q,a)
\end{array}
$$

L'automate A accepte le langage $\{[\,b^n\,]_1 \mid n \geq 0\}$. En effet, il suffit de remarquer que la transition δ_2 ne peut pas apparaître dans une exécution finie de A.

Nous construisons un automate B alternant élagué sur Γ_1 équivalent à A.

Pour toute transition $\delta = p,T \to R \in \Delta_A$, nous définissons la transition élaguée correspondante $\tilde{\delta} = p,T \to R \cap (Q_A \times \bar{\Gamma})$. Ainsi

$$\begin{array}{lll} \tilde{\delta}_1 = i \to (i,\bar{b}) & \tilde{\delta}_2 = i \to (i,\bar{a}) & \tilde{\delta}_3 = i,\bot_1 \to \emptyset \\ \tilde{\delta}_4 = p \to \emptyset & \tilde{\delta}_5 = p \to \emptyset & \tilde{\delta}_6 = q \to \emptyset \end{array}$$

Nous ne pouvons nous contenter de prendre $\{\tilde{\delta} \mid \delta \in \Delta_A\}$ comme ensemble de transitions de B. En effet, un tel automate accepterait $\{[\,w\,]_1 \mid w \in \{a,b\}^*\}$.

Pour décider quelles transitions élaguées doivent être conservées, il nous faut considérer le reste de la transition élaguée. Pour toute transition $\delta = p,T \to R$, nous définissons la transition $\delta' = p \to R \cap (Q_A \times \Gamma)$. Comme A est réduit, toute exécution de A commençant par δ' est entièrement étiquetée par Γ. Il est aisé de montrer que $\mathcal{S}_{\delta'}(A)$ est soit vide soit l'ensemble de toutes les piles sur Γ.

Pour pouvoir inclure $\tilde{\delta}$ dans l'ensemble des transitions de B, il nous suffit donc de vérifier que $\mathcal{S}_{\delta'}(A)$ est non vide. Dans notre cas, on vérifie facilement que seules δ'_2 et δ'_6 ne vérifient pas cette condition.

L'automate $B = (Q_A,I_A,\Delta_B)$ a pour ensemble de transitions:

$$\tilde{\delta}_1 = i \to (i,\bar{b}) \quad \tilde{\delta}_3 = i,\bot_1 \to \emptyset \quad \tilde{\delta}_4 = p \to \emptyset \quad \tilde{\delta}_5 = p \to \emptyset.$$

Par construction, $\mathcal{S}(B) \subseteq \mathcal{S}(A)$. Pour l'inclusion réciproque, il suffit de montrer que toute exécution acceptante de B peut être étendue en une exécution acceptante de A. Ceci est garanti par le fait que $\mathcal{S}'_\delta(A)$ est égale à $\mathrm{Stacks}_1(\Gamma)$. Cette construction est illustrée par la figure 4.4.

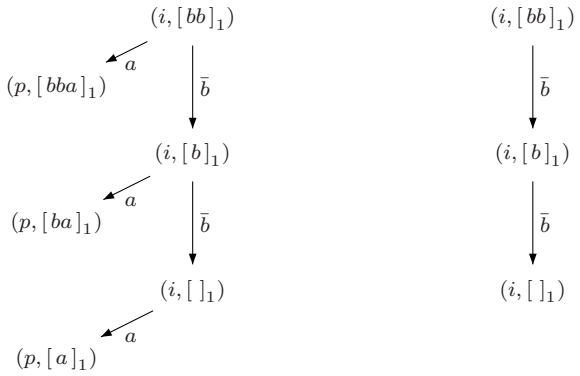


FIG. 4.4 – *Une exécution de A acceptant la pile $[bb]_1$ et l'exécution de B correspondante.*

Le lemme suivant permet de décider quelles transitions de l'automate élaguer dans l'automate alternant réduit sur Γ_1.

Lemme 4.3.29. *Pour tout automate $A = (Q_A,I_A,\Delta_A)$ alternant et réduit sur Γ_1 et pour toute transition $\delta \in \Delta_A$ telle que* $\mathrm{Test}(\delta) = \emptyset$ *et* $\pi_2(\mathrm{Act}(\delta)) \cap \overline{\Gamma} = \emptyset$, $\mathcal{S}_\delta(A)$ *est soit égal à* $\emptyset$ *soit à* $\mathrm{Stacks}_k(\Gamma)$. *De plus, nous pouvons en décider en* $\mathcal{O}(|A|)$.

Démonstration. Soit $A = (Q_A,I_A,\Delta_A)$ un automate réduit alternant sur Γ_1 et δ satisfaisant les conditions de l'énoncé. Comme nous ne nous intéressons qu'à $\mathcal{S}_\delta(A)$, nous pouvons supposer que pour tout $\delta \in \Delta_A$, $\mathrm{Test}(\delta) = \emptyset$ et $\pi_2(\mathrm{Act}(\delta)) \subseteq \Gamma$. Il suffit de montrer que si $\mathcal{S}_\delta(A)$ est non vide alors $\mathcal{S}_\delta(A)$ est égal à $\mathrm{Stacks}_k(\Gamma)$.

Supposons qu'il existe une pile s appartenant à Δ_A acceptée par une exécution $\mathcal{E}_A = (T_A,C_A)$ commençant par δ en s. Par définition d'une exécution, pour tout $u \in V_{T_A}$, la pile $\pi_2(C_A(u))$ étiquetant u est égale à $\mathcal{R}(\rho_u)(s)$ où $\rho_u \in \Gamma_1^*$ est telle que $r(T_A) \xrightarrow[T_A]{\rho_u} u$. Comme T_A n'est étiqueté que par Γ, il suit que ρ_u appartient à Γ^* et donc $\pi_2(C_A(u)) = s\rho_u$.
Soit $s' \in \mathrm{Stacks}_1(\Gamma)$. Nous définissons une exécution $\mathcal{E}'_A = (T_A,C'_A)$ acceptant s'. Pour tout $u \in V_{T_A}$, $C'_A(u) = (\pi_1(C_A(u)),s'\rho_u)$. Il est facile de vérifier que $\mathcal{E}'_A$ est bien une exécution de A et par construction, $\mathcal{E}'_A$ commence en s' par δ. Donc s' appartient à $\mathcal{S}_\delta(A)$.

Pour la partie effectivité, il suffit de tester le vide de $\mathcal{S}_\delta(A)$. D'après les restrictions syntaxiques sur A, cela revient à tester le vide d'un automate d'arbre *top-down* non-déterministe. Ce test peut être fait en $\mathcal{O}(|A|)$ (voir par exemple [CDG$^+$]). □

Nous pouvons maintenant établir que tout automate alternant réduit sur Γ_1 est équivalent à un automate réduit sur Γ_1.

Proposition 4.3.30. *Pour tout automate A alternant et réduit sur Γ_1, il existe un automate réduit sur Γ_1 tel que $\mathcal{S}(A) = \mathcal{S}(B)$. De plus, $|Q_B| \leq |\Gamma_k| \cdot (|Q_A|+1)$.*

Démonstration. Soit $A = (Q_A,I_A,\Delta_A)$ un automate alternant réduit sur Γ_1. Nous pouvons, sans perte de généralité, supposer que pour toute transition $\delta \in \Delta_A$, $|\pi_2(\mathrm{Act}(\delta)) \cap \overline{\Gamma}| \leq 1$. En effet, pour tout $x \neq y \in \Gamma$, $\mathrm{Dom}(\mathcal{R}(\bar{x})) \cap \mathrm{Dom}(\mathcal{R}(\bar{y})) = \emptyset$.

Nous élaguons A pour obtenir un automate $B = (Q_B,I_B,\Delta_B)$ élagué réduit sur Γ_1. Pour toute transition $\delta = p,T \to R \in \Delta_A$, nous définissons $\tilde{\delta} = p,T \to R \cap (Q_A \times \bar{\Gamma})$ et $\delta' = \bullet \to R \cap (Q_A \times \Gamma)$ où $\bullet$ est un symbole n'appartenant pas à Q_A. La transition δ' vérifie les conditions du lemme 4.3.29. Cependant δ' n'appartient pas à Δ_A. Soit $A_{\delta'}$ l'automate obtenu en rajoutant δ' à Δ_A.

Nous prenons $Q_B = Q_A$, $I_B = I_A$ et Δ_B est défini par:

$$\Delta_B = \{\tilde{\delta} \mid \delta \in \Delta_A \text{ et } \mathcal{S}_{\delta'}(A_{\delta'}) \neq \emptyset\}$$

Par construction B est élagué. Comme A est réduit, B l'est aussi. Par le lemme 4.3.29, B peut être construit en temps $\exp[0](|A|)$.

Par construction, $\mathcal{S}(B) \subseteq \mathcal{S}(A)$. Pour l'inclusion réciproque, il suffit de montrer que toute exécution acceptante de B peut être étendue en une exécution acceptante de A. Ceci est garanti par le fait que pour tout $\tilde{\delta} \in \Delta_B$, $\mathcal{S}'_\delta(A)$ est égale à $\mathrm{Stacks}_1(\Gamma)$ (cf. lemme 4.3.29).

D'après la proposition 4.3.24, il existe un automate C réduit sur Γ_1 équivalent à B avec $|Q_C| \leq |\Gamma_k|(|Q_B|+1)$. □

En combinant la proposition 4.3.25 et la proposition 4.3.30, il suit que tout automate alternant sur Γ_1 est équivalent à un automate réduit sur Γ_1.

Proposition 4.3.31. *Pour tout automate alternant A sur Γ_1, il existe un automate B réduit sur Γ_1 tel que $\mathcal{S}(B) = \mathcal{S}(A)$ et tel que $|Q_B|$ est bornée par* $\exp[1](|Q_A|)$.

Nous verrons dans ce paragraphe 4.4.2 que nous pouvons, avec la même complexité, transformer un automate alternant sur Γ_1 en un automate réduit déterministe sur Γ_1[7].

Par la proposition 4.3.14 et la proposition 4.3.31, il suit que $\mathrm{Rat}_1(\Gamma) = \mathrm{Alt}_1(\Gamma)$.

4.3.3 Automates sur Γ_k avec tests.

Pour étendre la proposition 4.3.30 à tous les niveaux, il nous faut introduire la notion d'automate avec tests.
Nous commençons par définir formellement les opérations de test. Pour tout $\ell \geq 1$ et $k \geq \ell$, l'opération de test de niveau k associée à un langage $L \subseteq \mathrm{Stacks}_\ell(\Gamma)$, notée Test^k_L, est définie pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ par:

$$\mathrm{Test}^k_T(s) = \begin{cases} s & \text{si } \mathrm{top}_\ell(s) \in L, \\ \text{non défini} & \text{sinon.} \end{cases}$$

En d'autres termes, l'opération Test^k_L est l'identité de niveau k restreinte à l'ensemble des piles de niveau k dont la dernière pile de niveau ℓ appartient à L. Nous définissons naturellement l'instruction correspondante notée T^k_L et nous étendons $\mathcal{R}$ en prenant $\mathcal{R}(T^k_L) = \mathrm{Test}^k_L$. Quand k se déduit du contexte, nous écrivons simplement Test_L et T_L.

Un automate sur Γ_k avec tests dans un ensemble fini $\mathcal{L}$ de parties de $\mathrm{Stacks}_\ell(\Gamma)$ (pour $\ell \leq k$) est défini en ajoutant, aux tests de fonds de piles, des instructions de test dans $\mathcal{T}^k_\mathcal{L} = \{T^k_L \mid L \in \mathcal{L}\}$. Pour tout $T \subseteq \mathcal{T}^k_\mathcal{L}$, nous définissons $\mathrm{Dom}(T) \subseteq \mathrm{Stacks}_\ell$ par:

$$\mathrm{Dom}(T) = \begin{cases} \bigcap_{\ell \in [1,n]} L_\ell & \text{si } T = \{ T^k_{L_1}, \ldots, T^k_{L_n} \} \text{ avec } n > 0, \\ \mathrm{Stacks}_\ell(\Gamma) & \text{si } T = \emptyset \end{cases}$$

7. Rappelons qu'un automate réduit sur Γ_1 est simplement un automate fini sur Γ. La notion de déterminisme est donc la notion classique pour les automates finis sur le monoïde libre Γ^*.

Définition 4.3.32. Un automate A sur Γ_k avec tests dans un ensemble fini $\mathcal{L}$ de parties de $\text{Stacks}_\ell(\Gamma)$ où $\ell \leq k$ est un quadruplet (Q,I,F,Δ) où Q est l'ensemble fini des états, $I \subseteq Q$ et $F \subseteq Q$ sont respectivement les ensembles des états initiaux et finaux, et $\Delta \subseteq Q \times \Gamma_k^{\text{o}} \times \text{Sing}(\Gamma_k^{\text{t}}) \times 2^{\mathcal{T}_{\mathcal{L}}^k} \times Q$ est l'ensemble des transitions.

Nous noterons $p \xrightarrow{\gamma} q,T,T'$ la transition $(p,\gamma,T,T',q) \in \Delta$. Intuitivement dans une configuration (p,s) l'automate A peut appliquer la transition $p \xrightarrow{\gamma} q,T,T'$ pour passer dans la configuration (q,s') si la pile $s' = \mathcal{R}(\gamma)(s)$ est définie et si s' appartient à $\text{Dom}(T')$

La notion de langage accepté est définie de manière analogue au cas des automates dans Γ_k. Nous pouvons définir de la même manière la notion d'automate alternant sur Γ_k avec tests dans $\mathcal{L}$. De même les notions d'automates réduits et élagués se transposent immédiatement en présence des tests.

Exemple 4.3.33. Considérons le langage S de piles de niveau 2 sur $\Gamma = \{a,b\}$ défini par:

$$\{[\,[\,w_0\,]\,[\,w_1\,]\ldots[\,w_n\,]\,]_2 \quad | \quad w_0 \in \Gamma^*, n > 0 \text{ et pour tout } i \in [1,n], \\ w_i \sqsubseteq w_{i-1} \text{ et } |w_i|_a \mod 2 = |w_i|_b \mod 2 = 0\}.$$

Ainsi les piles $[\,[\,abbaaabb\,]\,[\,abbaaa\,]\,[\,abba\,]\,[\,]\,]_2$ et $[\,[\,abbaaabb\,]\,[\,abba\,]\,]_2$ appartiennent à S. Il est facile de vérifier que S appartient à Rat_2. Nous donnons un automate $A = (Q,I,F,\Delta)$ réduit sur Γ_2 avec tests dans $\{A,B\}$ où A et B sont les langages de piles de niveau 1 respectivement définis par $\{w \in \Gamma^* \mid |w|_a \mod 2 = 0\}$ et $\{w \in \Gamma^* \mid |w|_b \mod 2 = 0\}$ acceptant S. L'automate A est représenté dans la figure 4.5.

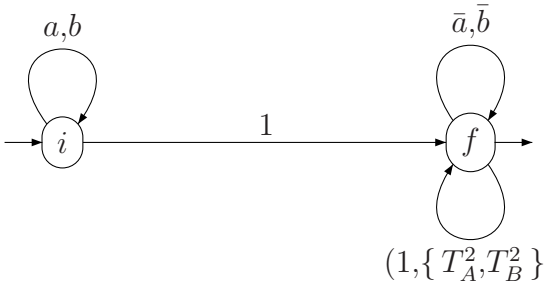


FIG. 4.5 – *Un automate réduit sur* Γ_2 *avec tests dans* $\{A,B\}$ *acceptant* S.

Pour des raisons liées à la complexité des transformations, nous allons ajouter au uplet définissant l'automate $A = (Q,I,F,\Delta)$ une application μ de I dans $2^{\mathcal{T}_{\mathcal{L}}^k}$. La sémantique de cette application est qu'un état initial $i \in I$ peut être l'état de départ d'un calcul acceptant de A si et seulement si la pile vide $[\,]_k$ appartient à $\text{Dom}(\mathcal{R}(t))$ pour tout $t \in \mu(i)$. Du point de vue de l'expressivité, cet ajout n'a

aucune incidence. En effet, nous pouvons supprimer μ sans changer le langage accepté, en prenant l'ensemble $\{i \in I \mid [\,]_k \in \mathrm{Dom}(\mathcal{R}(t)) \text{ pour tout } t \in \mu(i)\}$ comme nouvel ensemble d'états initiaux. Cependant, le coût de cette transformation est lié au coût du test de l'appartenance de la pile vide $[\,]_\ell$ aux langages de $\mathcal{L}$. L'ajout de l'application μ permet d'éviter de faire apparaître ce coût dans les transformations présentées dans la suite du chapitre.

Nous prolongeons la correspondance entre les automates alternants élagués sur Γ_k et les automates sur Γ_k en présence de tests. La preuve est une adaptation immédiate de la preuve de la proposition 4.3.24.

Proposition 4.3.34. *Les automates alternants élagués sur Γ_k avec tests dans un ensemble fini $\mathcal{L}$ de parties de $\mathrm{Stacks}_\ell(\Gamma)$ avec $\ell \leq k$ et les automates réduits sur Γ_k avec tests dans $\mathcal{L}$ sont équivalents:*

1. *Pour tout automate A alternant élagué (resp. élagué et réduit) sur Γ_k avec tests dans $\mathcal{L}$, il existe un automate B (resp. réduit) sur Γ_k avec tests dans $\mathcal{L}$ tel que $\mathcal{S}(B) = \mathcal{S}(A)$ et $|Q_B| = |Q_A|$.*
2. *Pour tout automate A (resp. réduit) sur Γ_k, il existe un automate B alternant élagué (resp. élagué et réduit) sur Γ_k tel que $\mathcal{S}(B) = \mathcal{S}(A)$ et $.|Q_B| \leq |\Gamma_k|(|Q_A|+1)$.*

Remarque 4.3.35. Grâce à l'ajout de l'application μ, cette équivalence est effective sans présumer de la décidabilité de l'appartenance de la pile vide aux langages de $\mathcal{L}$.

Si nous enrichissons un automate sur Γ_{k+1} avec des tests dans $\mathrm{Rat}_k(\Gamma)$, nous n'augmentons pas l'expressivité du modèle. En effet, soient $R \in \mathrm{Rat}_k$ et $I \in \mathrm{Rat}(\Gamma_k^*)$ telle que $R = \mathcal{R}(I)([\,]_k)$, il est facile de vérifier que:

$$\mathrm{Test}_R^{k+1} = \mathrm{copy}_k \cdot \mathcal{R}(\bar{I} \cdot \perp_k \cdot (\Gamma_k^{\mathrm{o}})^*) \cdot \overline{\mathrm{copy}}_k. \tag{4.7}$$

Proposition 4.3.36. *Pour tout automate A sur Γ_{k+1} avec tests dans un ensemble fini $\mathcal{L} \subseteq Rat_k$, il existe un automate B sur Γ_{k+1} tel que $\mathcal{S}(B) = \mathcal{S}(A)$. De plus si chaque $L \in \mathcal{L}$ est accepté par un automate $A_L \in Rat(\Gamma_k)$ alors nous pouvons supposer que $|Q_B|$ est bornée par $\exp[0](|A| + \sum_{L\in\mathcal{L}} |A_L|)$.*

Démonstration. Soit $A = (Q,I,F,\Delta,\mu)$ un automate sur Γ_{k+1} avec tests dans un sous-ensemble fini $\mathcal{L}$ de $\mathrm{Rat}_k(\Gamma)$. Pour tout $L \in \mathcal{L}$, nous notons I_L un ensemble de $\mathrm{Rat}(\Gamma_k)$ tel que $L = \mathcal{R}(I_L)([\,]_k)$. Pour toute transition $\delta = p \xrightarrow{\gamma} q,T,T' \in \Delta$, nous définissons un ensemble rationnel d'instructions de Γ_{k+1}, noté R_δ, par:

$$\begin{cases} R_\delta &= \{\varepsilon\} & \text{si } T' = \emptyset, \\ R_\delta &= k\overline{I}_{L_1} \perp_k \Gamma_k^* \bar{k} \cdots k\overline{I}_{L_n} \perp_k \Gamma_k^* \bar{k} & \text{si } T' = \{\, T_{L_1}, \ldots, T_{L_n} \,\}. \end{cases}$$

De la même manière, nous définissons pour tout état initial $p \in I$ un ensemble de $\mathrm{Rat}(\Gamma_{k+1}^*)$ associé à l'ensemble $\mu(p)$ et noté R_p.

Soit $A_\delta = (Q_\delta, I_\delta, F_\delta, \Delta_\delta)$ un automate sur Γ_{k+1} acceptant le langage des suites d'instructions R_δ (*i.e.* $\mathcal{I}(A_\delta) = R_\delta$). Nous pouvons, sans perte de généralité, supposer que A possède un unique état initial i_δ et un unique état final f_δ (en utilisant des ε-transitions). De la même manière, nous prenons pour tout état initial $p \in I$ un automate $A_p = (Q_p, I_p, F_p, \Delta_p)$ acceptant R_i et avec un unique état initial i_p et un unique état final f_p.

Nous supposons aussi que les ensembles Q_δ et les Q_p sont deux-à-deux disjoints et disjoints de Q.

Nous construisons maintenant l'automate $B = (Q_B, I_B, F_B, \Delta_B)$ sur Γ_{k+1} en prenant:

$$\begin{array}{lcl} Q_B & = & Q_A \cup \bigcup_{\delta\in\Delta} Q_\delta \cup \bigcup_{p\in I} Q_i \\ I_B & = & \{i_p \mid p \in I\} \\ F_B & = & F \\ \Delta_B & = & \bigcup_{\delta\in\Delta} \Delta_\delta \cup \bigcup_{p\in I} \Delta_p, \\ & \cup & \{f_p \xrightarrow{\varepsilon} p \mid p \in I\}, \\ & \cup & \{p \xrightarrow{\gamma} i_\delta, T \ , \ f_\delta \xrightarrow{\varepsilon} q \mid \delta = p \xrightarrow{\gamma} q, T, T' \in \Delta\}. \end{array}$$

De l'équation (4.7), il suit que $\mathcal{S}(B) = \mathcal{S}(A)$. □

Exemple 4.3.37. Nous illustrons dans cet exemple le résultat de la proposition précédente sur l'automate A présenté dans l'exemple 4.3.33. Cet automate réduit sur Γ_2 avec tests dans $\{A, B\} \subset \mathrm{Rat}_1(\Gamma)$ où A et B sont les langages de piles de niveau 1 respectivement définis par $\{w \in \Gamma^* \mid |w|_a \mod 2 = 0\}$ et $\{w \in \Gamma^* \mid |w|_b \mod 2 = 0\}$ accepte le langage:

$$\begin{array}{rcl} S & = & \{[\,[\,w_0\,]\,[\,w_1\,]\ldots[\,w_n\,]\,]_2 \mid w_0 \in \Gamma^*, n > 0 \text{ et pour tout } i \in [1,n], \\ & & \qquad w_i \sqsubseteq w_{i-1} \text{ et } |w_i|_a \mod 2 = |w_i|_b \mod 2 = 0\}. \end{array}$$

D'après la proposition 4.3.36, le langage S appartient à $\mathrm{Rat}_2(\Gamma)$. La figure 4.6 présente un automate B sur Γ_2 acceptant S.

Nous pouvons déduire un résultat plus fort pour les automates alternants : les automates alternants sur Γ_k avec tests dans Alt_k sont équivalents aux automates alternants sur Γ_k.

Proposition 4.3.38. *Pour tout automate A alternant sur Γ_k avec tests dans un ensemble fini $\mathcal{L} \subseteq Alt_k$, il existe un automate B alternant sur Γ_k tel que $\mathcal{S}(B) = \mathcal{S}(A)$. De plus si chaque $L \in \mathcal{L}$ est accepté par un automate A_L alternant sur Γ_k alors nous pouvons supposer que $|Q_B|$ est bornée par* $\exp[0](|Q_A| + \sum_{L\in\mathcal{L}} |A_L|)$.

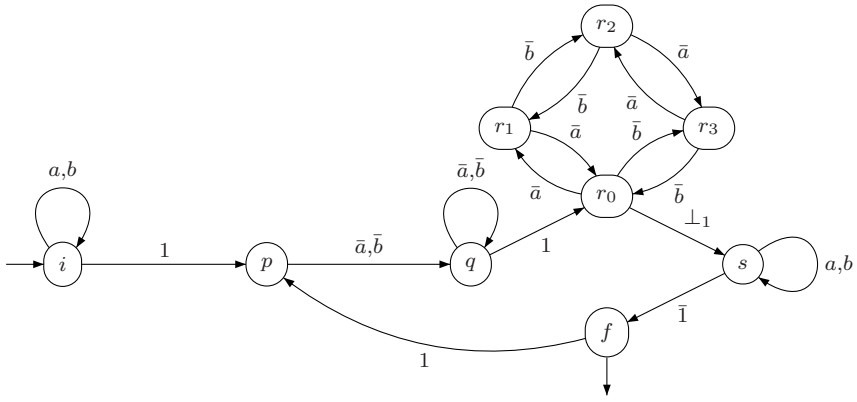


FIG. 4.6 – *Un automate B sur Γ_2 acceptant le langage S présenté dans l'exemple 4.3.33.*

Démonstration. Soit $A = (Q_A,I_A,\Delta_A)$ un automate alternant sur Γ_k avec tests dans un ensemble fini $\mathcal{L} \subset \mathrm{Alt}_k$. De plus, supposons que chaque langage $L \in \mathcal{L}$ soit accepté par un automate $A_L = (Q_L,I_L,\Delta_L)$ alternant sur Γ_k. Nous pouvons supposer, sans perte de généralité, que les ensembles d'états de tous ces automates sont deux à deux disjoints.

Nous allons construire un automate $B = (Q_B,I_B,\Delta_B)$ alternant sur Γ_k acceptant $\mathcal{S}(A)$.

L'ensemble des états Q_B est $Q_A \cup \bigcup_{L\in\mathcal{L}} Q_L$, les états initiaux I_B sont les états initiaux de A et l'ensemble des transitions Δ_B est donné par:

$$\begin{array}{rcl} \Delta_B & = & \{p,T \to R \wedge \bigwedge_{T_L^k \in T'}(i_L,\varepsilon) \mid \delta = p,T,T' \to R \in \Delta_A \text{ et } i_L \in I_L\} \\ & \cup & \bigcup_{L\in\mathcal{L}} \Delta_L \end{array}$$

Par construction, B est équivalent à A. Remarquons qu'en général, $|B|$ est exponentiel en $|Q_B|$. □

4.3.3.1 Équivalence entre automates alternants et automates non alternants.

Dans ce sous-paragraphe, nous établissons que tout automate alternant sur Γ_k est équivalent à un automate sur Γ_k.

L'étape clé est d'établir que tout automate réduit sur Γ_{k+1} est équivalent à un automate réduit sur Γ_{k+1} avec tests dans Alt_k. L'intuition de la preuve est la même que pour le niveau 1 (cf. sous-paragraphe 4.3.2.2).
Nous élaguons les transitions de l'automate $A = (Q_A,I_A,\Delta_A)$ alternant et réduit sur Γ_{k+1}. Pour toute transition $\delta = p,T \to R \in \Delta_A$ et pour tout $\gamma \in \Gamma^{\mathrm{o}}_{k+1}$,

nous définissons la transition élaguée $\tilde{\delta}_\gamma = p,T \to R \cap (Q_A \times \{\gamma\})$ et le reste de l'élagage $\delta_\gamma = \bullet \to R \cap (Q_A \times (\Gamma^{\circ}_{k+1} \setminus \{\gamma\}))$.

Au niveau 1, l'ensemble $\mathcal{S}_{\delta_\gamma}(A)$ était soit vide soit égal à $\mathrm{Stacks}_1(\Gamma)$. Aux niveaux supérieurs, cette propriété n'est plus vérifiée. Nous allons montrer qu'il existe un automate alternant $A_{\delta,\gamma}$ de niveau k tel que pour toute pile $s = [\, s_1 \dots s_n \,]_{k+1}$ avec $\mathrm{Last}(s) = \bar{\gamma}$, $s \in \mathcal{S}_{\delta_\gamma}(A)$ si et seulement si $s_n \in \mathcal{S}(A_{\delta,\gamma})$.

Ces automates et leurs propriétés sont donnés par les deux lemmes ci-dessous.

Lemme 4.3.39. *Pour tout $k \geq 1$, tout automate alternant réduit sur Γ_{k+1}, pour tout $\gamma \in \Gamma^{\circ}_{k+1}$ et toute transition $\delta \in \Delta_A$ telle que $\bar{\gamma} \notin \pi_2(\mathrm{Act}(\delta))$ et telle que $\mathrm{Test}(\delta) = \emptyset$, il existe un automate $A_{\delta,\gamma}$ alternant sur Γ_k tel que pour toute pile $s = [\, s_1, \dots, s_n \,]_{k+1}$ avec $Last(s) = \gamma$,*

$$s \in \mathcal{S}_\delta(A) \quad \Leftrightarrow \quad s_n \in \mathcal{S}(A_{\delta,\gamma}).$$

De plus $|Q_{A_{\delta,\gamma}}|$ et $|A_{\delta,\gamma}|$ sont respectivement bornés par $\exp[0](|Q_A|)$ *et* $\exp[0](|A|)$.

Démonstration. Soit A un automate alternant réduit sur Γ_{k+1}, γ un symbole de Γ°_{k+1} et δ_0 une transition de Δ_A tels que $\bar{\gamma} \notin \pi_2(\mathrm{Act}(\delta_0))$ et $\mathrm{Test}(\delta_0) = \emptyset$.

Nous commençons par montrer que nous pouvons supprimer toutes les transitions de Δ_A contenant l'instruction $\bar{k}$ et l'instruction $\bot_{k+1}$ sans changer $\mathcal{S}_{\delta_0}(A) \cap \{s \in \mathrm{Stacks}_{k+1}(\Gamma) \mid \mathrm{Last}(s) = \gamma\}$.

Soit $s = [\, s_1, \dots, s_n \,] \in \mathrm{Stacks}_{k+1}(\Gamma)$ de suite réduite ρ_s avec $\rho_s(|\rho_s|) = \gamma$ (*i.e.* $\mathrm{Last}(s) = \gamma$). Supposons que s soit acceptée par une exécution $\mathcal{E}_A(T_A,C_A)$ de A commençant en s par δ_0 (*i.e.* $s \in \mathcal{S}_{\delta_0}(A)$). Montrons que $\bar{k}$ n'apparaît pas comme étiquette de T_A.

Soit $u \in V_{T_A}$ avec $C_A(u) = (p,s')$. Il existe $\rho' \in (\Gamma^{\circ}_{k+1})^*$ tel que $r(T_A) \xrightarrow[A]{\rho'} u$. Par définition d'une exécution, $s' = \mathcal{R}(\rho')(s)$. Donc $s' = \mathcal{R}(\rho_s\rho')([\,]_{k+1})$. La suite $\rho_s\rho'$ est réduite. En effet ρ' est réduite car A est réduit, ρ_s est réduite par définition et, par définition de δ, $\rho'(1) \neq \bar{\gamma} = \overline{\rho_s(1)}$. Donc $\rho_s\rho'$ est la suite réduite de s' et par la remarque 4.1.11, elle ne peut pas contenir d'occurrence du symbole $\bar{k}$. Il suit que $\bar{k}$ ne peut apparaître comme étiquette de T_A. Par un raisonnement analogue, nous établissons que $[\,]_{k+1}$ n'apparaît pas dans $\pi_2(C_A(V_{T_A}))$

Nous supposons maintenant que pour toute transition $\delta \in \Delta_A$, l'instruction $\bar{k}$ n'apparaît pas dans $\pi_2(\mathrm{Act}(\delta))$ et que l'instruction $\bot_{k+1}$ n'apparaît pas dans Δ_A.

Nous allons construire un automate $B = (Q_A \cup \{i_0\}, \{i_0\}, \Delta_B)$ où i_0 est un symbole n'appartenant pas à Q_A. L'ensemble Δ_B est obtenu en remplaçant dans Δ_A toutes les occurrences de l'instruction k par ϵ.

$$\begin{array}{rcl} \Delta_B & = & \{p,T \to (p_1,\tilde{\gamma_1}) \wedge \dots \wedge (p_n,\tilde{\gamma_n}) \mid p,T \to (p_1,\gamma_1) \wedge \dots \wedge (p_n,\gamma_n) \in \Delta_A\} \\ & \cup & \{i_0,T \to (p_1,\tilde{\gamma_1}) \wedge \dots \wedge (p_n,\tilde{\gamma_n}) \mid \delta_0 = p,T \to (p_1,\gamma_1) \wedge \dots \wedge (p_n,\gamma_n)\} \end{array}$$

où pour tout $\gamma \in \Gamma_k^{\circ}$, $\tilde{\gamma} = \gamma$ et $\tilde{k} = \varepsilon$.

Nous obtenons donc par construction un automate alternant sur Γ_k avec des ε-transitions. Par construction et par le lemme 4.1.3, il en résulte que pour toute pile $s = [\, s_1, \ldots, s_n \,]_{k+1}$ avec $\mathrm{Last}(s) = \gamma$, $s \in \mathcal{S}_{\delta_0}(A)$ si et seulement si $s_n \in \mathcal{S}(A_{\delta_0})$.

Remarquons qu'il ne faut pas éliminer les ε-transitions à cette étape. En effet, l'élimination des ε-transitions peut produire un automate A_{δ_0} de taille exponentielle par rapport à $|A|$. □

Le lemme précédent traite le cas des piles non vides (*i.e.* $\mathrm{Last}(s)$ est défini). Le lemme suivant traite le cas de la pile vide du niveau $k+1$. Sa preuve est une adaptation immédiate de la preuve du lemme précédent.

Lemme 4.3.40. *Pour tout $k \geq 1$, tout automate alternant réduit sur Γ_{k+1}, et toute transition $\delta \in \Delta_A$, il existe un automate $A_{\delta,\varepsilon}$ alternant sur Γ_k tel que:*

$$[\,]_{k+1} \in \mathcal{S}_{\delta}(A) \quad \Leftrightarrow \quad [\,]_k \in \mathcal{S}(A_{\delta,\varepsilon}).$$

De plus $|Q_{A_{\delta,\varepsilon}}|$ et $|A_{\delta,\varepsilon}|$ sont respectivement bornés par $\exp[0](|Q_A|)$ *et* $\exp[0](|A|)$.

En utilisant les lemmes précédents, nous pouvons construire à partir d'un automate alternant réduit sur Γ_{k+1}, un automate réduit sur Γ_{k+1} avec tests dans Alt_k équivalent.

Proposition 4.3.41. *Pour tout $k \geq 1$, tout automate A sur Γ_{k+1} alternant et réduit est équivalent à un automate B réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset Alt_k$. De plus, la taille Δ_B est polynomiale en la taille Δ_A, $|\mathcal{L}| \leq (k+2) \cdot |\Delta_A|$ et tout $L \in \mathcal{L}$ est accepté par un automate A_L alternant sur Γ_k avec $|Q_{A_L}|$ et $|A_L|$ respectivement bornés par* $\exp[0](|Q_A|)$ *et* $\exp[0](|A|)$.

Démonstration. Soit $A = (Q_A, I_A, \Delta_A)$ un automate sur Γ_{k+1} alternant et réduit. Nous construisons un automate B alternant élagué et réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset \mathrm{Alt}_k$.

Pour tout $\delta = p, T \to R \in \Delta_A$ et tout $\gamma \in \Gamma_{k+1}^{\circ}$, nous définissons $\tilde{\delta}_{\gamma} = p, R \to R \cap (Q_A \times \{\, \gamma \,\})$ et $\delta_{\gamma} = \bullet \to R \cap (Q_A \times (\Gamma_k^{\circ} \setminus \{\, \gamma \,\}))$ où $\bullet$ est un symbole n'appartenant pas à Q_A. Nous noterons $A_{\delta_{\gamma}}$ l'automate alternant sur Γ_k construit dans le lemme 4.3.39 pour l'automate $(Q_A, I_A, \Delta_A \cup \{\delta_{\gamma}\})$.
De même, pour toute transition $\delta = p, T \to R \in \Delta_A$, nous définissons $\delta_{\varepsilon} = \bullet \to R$ et notons $A_{\delta_{\varepsilon}}$ l'automate alternant sur Γ_k construit dans le lemme 4.3.40.

Enfin, l'ensemble $\mathcal{L} \subset \mathrm{Alt}_k$ des langages de test est:

$$\mathcal{L} = \{\mathcal{S}(A_{\delta_{\gamma}}), \mathcal{S}(A_{\delta_{\varepsilon}}) \mid \delta \in \Delta_A \text{ et } \gamma \in \Gamma_{k+1}^{\circ}\}.$$

Nous définissons maintenant un automate $B = (Q_B, I_B, \Delta_B)$ alternant élagué et réduit sur Γ_{k+1} avec tests dans $\mathcal{L}$. Nous prenons $Q_B = (Q_A \cup \{\, \bullet \,\}) \times (\Gamma_k^{\circ} \cup$

$\{\varepsilon,k\}$), $I_B = I_A \times (\Gamma_k^\circ \cup \{k,\varepsilon\})$ et l'ensemble des transitions est défini par:

$$\begin{array}{rcl}
\Delta_B & = & \{(p,\gamma),T,T_{\mathcal{S}(A_{\delta_{\bar\gamma}})} \to ((q,\gamma'),\bar\gamma) \mid \delta \in \Delta_A \text{ et } \tilde\delta_{\bar\gamma} = p,T \to (q,\bar\gamma)\} \\
& \cup & \{(p,\gamma),T,T_{\mathcal{S}(A_{\delta_{\bar\gamma}})} \to ((\bullet,\gamma'),\bar\gamma) \mid \delta \in \Delta_A \text{ et } \tilde\delta_{\bar\gamma} = p,T \to \emptyset\} \\
& \cup & \{(p,\varepsilon),\bot_{k+1},T_{\mathcal{S}(A_{\delta_\varepsilon})} \to \emptyset \mid \delta \in \Delta_A \text{ et } \mathrm{Head}(\delta) = p\} \\
& \cup & \{(\bullet,\gamma) \to ((\bullet,\gamma'),\bar\gamma)\} \\
& \cup & \{(\bullet,\varepsilon),\bot_{k+1} \to \emptyset\}
\end{array}$$

où dans tous les ensembles, $\gamma \in \Gamma_k^\circ \cup \{k\}$ et $\gamma' \in \Gamma_k^\circ \cup \{k,\varepsilon\}$ avec $\gamma' \neq \bar\gamma$.

Par construction, B est un automate alternant élagué réduit sur Γ_{k+1} et $\mathcal{S}(A) \subseteq \mathcal{S}(B)$.

Pour l'inclusion réciproque, nous montrons que toute exécution acceptante $\mathcal{E}_B$ de l'automate B peut s'étendre en une exécution acceptante de A.

Avant d'établir cette propriété, remarquons que pour toute exécution (non nécessairement acceptante) $\mathcal{E}_A = (T_A,C_A)$ de A si $C_A(r(T_A) = ((q,\gamma),s)$ alors $\mathrm{Last}(s) = \gamma$. Une récurrence immédiate sur la longueur de l'exécution $\mathcal{E}_A$, utilisant les lemmes 4.3.39 et 4.3.40 établit que:
pour toute exécution $\mathcal{E}_B$ de B commençant par l'état $(q,\gamma) \in Q_A \times (\Gamma_{k+1}^\circ \cup \{\varepsilon\})$ sur la pile s, il existe une exécution $\mathcal{E}_A$ de A commençant par q sur la pile s. Il s'en suit donc que $\mathcal{S}(B) \subseteq \mathcal{S}(A)$.
Nous avons donc établi l'existence d'un automate alternant élagué et réduit avec test dans $\mathcal{L} \subset \mathrm{Alt}_k(\Gamma)$ équivalent à A. D'après la proposition 4.3.24, il existe un automate C réduit sur Γ_{k+1} avec tests dans $\mathcal{L}$ équivalent à B. □

Un corollaire immédiat de cette proposition et de la proposition 4.3.25 est que tout automate alternant sur Γ_{k+1} est équivalent à un automate réduit sur Γ_{k+1} avec tests dans Alt_k.

Corollaire 4.3.42. *Pour tout $k \geq 1$, tout automate A alternant sur Γ_{k+1} est équivalent à un automate B réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset Alt_k(\Gamma)$. De plus, $|Q_B|$ et $|\mathcal{L}|$ sont bornées par* $\exp[1](|Q_A|)$ *et chaque langage $L \in \mathcal{L}$ est accepté par un automate A_L alternant sur Γ_k de taille bornée par* $\exp[1](|Q_A|)$.

Nous pouvons maintenant établir, par récurrence sur le niveau k, que tout automate alternant sur Γ_k est équivalent à un automate sur Γ_k.

Théorème 4.3.43. *Pour tout automate alternant A sur Γ_k, il existe un automate B sur Γ_k équivalent. De plus, nous pouvons supposer que la taille de B est bornée par* $\exp[k](|Q_A|)$.

Démonstration. La preuve procède par récurrence sur le niveau k.
Cas de base : $k = 1$**.** Ce cas a été démontré dans la proposition 4.3.31.

Etape de récurrence. Soit $A = (Q_A,I_A,\Delta_A)$ un automate alternant sur Γ_{k+1}. Par la proposition 4.3.25, il existe un automate $B = (Q_B,I_B,\Delta_B)$ alternant réduit sur Γ_{k+1} équivalent à A. Par la proposition 4.3.41, il existe un automate C réduit sur Γ_k avec tests dans $\mathcal{L} \subset \mathrm{Alt}_k$. Comme par hypothèse de récurrence, tout automate alternant sur Γ_k est équivalent à un automate sur Γ_k, C est un automate réduit sur Γ_{k+1} avec tests dans Rat_k. Par la proposition 4.3.36, il existe un automate D sur Γ_{k+1} équivalent à C.

Un rapide calcul des complexités mises en jeu dans chaque transformation montre que $|Q_D|$ est bornée par $\exp[k+1](|Q_A|)$. □

Il en résulte immédiatement que pour tout $k \geq 1$,

$$\mathrm{Alt}_k = \mathrm{Rat}_k.$$

Un corollaire du théorème précédent et des propositions 4.3.34 et 4.3.38 est que les langages acceptés par les automates sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ appartiennent à $\mathrm{Rat}_k(\Gamma)$.

Corollaire 4.3.44. *Pour tout $k \geq 1$, $Rat_k(\Gamma) = \mathcal{R}(Rat((\Gamma_k \cup \mathcal{T}_{Rat_k(\Gamma)})^*))([\ ]_k)$.*

4.3.3.2 Première normalisation

Dans ce sous-paragraphe, nous établissons la première normalisation des automates sur Γ_{k+1}. Plus précisément, nous montrons que tout automate sur Γ_{k+1} est équivalent à un automate réduit sur Γ_{k+1} avec tests dans Rat_k.

Ce résultat peut être déduit des résultats du sous-paragraphe précédent. En effet, comme tout automate sur Γ_{k+1} est équivalent à un automate alternant sur Γ_{k+1} (cf. proposition 4.3.14), il découle de la proposition 4.3.41 et du théorème 4.3.43 que tout automate sur Γ_{k+1} est équivalent à un automate réduit sur Γ_{k+1} avec tests dans Rat_k. Cependant du point de vue de la complexité, les automates sur Γ_k acceptant les langages de tests ont une taille $(k+1)$ fois exponentielle en la taille de l'automate de départ. Nous allons montrer que, par une construction *ad hoc*, nous pouvons réduire d'un niveau la taille de la tour d'exponentielles.

Intuitivement, cette construction consiste à éliminer les boucles dans les exécutions de l'automate A sur Γ_{k+1}. Nous dirons que l'automate A boucle sur la pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ en partant dans l'état p et en revenant dans l'état q s'il existe un calcul $(p_0,s_0) \xrightarrow[A]{} \cdots \xrightarrow[A]{} (p_n,s_n)$ tel que $s_0 = s_n = s$, $p_0 = p$, $p_n = q$ et pour tout $\ell \in [0,n]$, $\rho_s \sqsubseteq \rho_{s_\ell}$. Pour tout $p,q \in Q_A$, nous noterons $L_{p,q} \subseteq \mathrm{Stacks}_{k+1}(\Gamma)$ l'ensemble des piles s tel que A peut boucler sur s en partant en p et en revenant en q.

Exemple 4.3.45. Considérons l'automate A sur Γ_2 (avec $\Gamma = \{a,b\}$) présenté dans la figure 4.1 p. 98. Les langages des boucles de cet automate sont donnés ci-dessous:

$$\begin{array}{lcl} L_{p_1,q_1} & = & \{[\,s_1,\ldots,s_n\,]_2 \mid s_n(|s_n|) = a\} \\ L_{p_2,q_2} & = & \{[\,s_1,\ldots,s_n\,]_2 \mid s_n(|s_n|) = b\} \\ L_{p,q} & = & \mathrm{Stacks}_2(\Gamma) \\ L_{i,r} & = & \mathrm{Stacks}_2(\Gamma) \end{array}$$

Tous les autres langages de boucles sont vides. Nous retrouvons bien que l'ensemble des piles acceptées par A est $\{[\,[\,w\,]\,]_2 \mid w \in \Gamma^*\}$.

Nous allons montrer, dans le lemme suivant, que l'appartenance d'une pile s de niveau $k+1$ à $L_{p,q}$ ne dépend que de $\mathrm{Last}(s)$ et de $\mathrm{top}_k(s)$ la dernière pile de niveau k de s.

Lemme 4.3.46. *Pour tout automate A sur Γ_{k+1}, tous $p,q \in Q_A$ et $\gamma \in \Gamma^{\mathrm{o}}_{k+1} \cup \{\varepsilon\}$, il existe un automate $A^{\gamma}_{p,q}$ alternant sur Γ_k tel que pour toute pile $s = [\,s_1,\ldots,s_n\,]_{k+1}$ avec $\mathrm{Last}(s) = \gamma$, on a $s \in L_{p,q}$ si et seulement si $s_n \in \mathcal{S}(A^{\gamma}_{p,q})$.*

Démonstration. Soit $A = (Q_A,I_A,F_A,\mu_A,\Delta)$ un automate sur Γ_{k+1}, $\gamma_0 \in \Gamma^{\mathrm{o}}_k \cup \{k,\varepsilon\}$ et $p_0,q_0 \in Q_A$.

Considérons le cas $\gamma_0 \neq \varepsilon$. Le cas $\gamma_0 = \varepsilon$ est similaire.

Comme $\mathrm{Last}(s) \neq \varepsilon$, la pile s est non vide. Toutes les piles intervenant dans une boucle de A sur s sont différentes de la pile vide de niveau $k+1$. Nous pouvons donc supprimer toutes les transitions de A contenant l'instruction de test $\bot_{k+1}$.

Nous commençons par construire un automate $B = (Q_B,I_B,\Delta_B)$ alternant réduit sur Γ_{k+1} acceptant $L_{p_0,q_0} \cap \{s \in \mathrm{Stacks}_k(\Gamma) \mid \mathrm{Last}(s) = \gamma_0\}$. L'ensemble des états Q_B est égal à $Q_A \times Q_A \times (\Gamma^{\mathrm{o}}_k \cup \{k\}) \times \mathrm{Sing}(\Gamma^{\mathrm{T}}_k)$ et l'ensemble des états initiaux I_B est $\{(p_0,q_0,\gamma_0,T) \mid T \in \mathrm{Sing}(\Gamma^{\mathrm{T}}_k)\}$.

Intuitivement, si une configuration $((p,q,\gamma,T),s)$ apparaît dans une exécution acceptante de B commençant sur une pile s_0 telle que $\mathrm{Last}(s_0) = \gamma_0$ alors:

- $\mathrm{Last}(s) = \gamma$,
- $s \in \mathrm{Dom}(\mathcal{R}(t))$ pour tout $t \in T$,
- et s appartient à $L_{p,q}$.

Pour définir l'ensemble des transitions Δ_B, nous considérons des transitions δ de la forme suivante:

$$\delta = (p,q,\gamma,T),T \rightarrow \bigwedge_{\gamma' \in R} \quad \bigwedge_{(p',q',\gamma',T') \in Q_{\gamma'}} ((p',q',\gamma',T'),\gamma')$$

où $R \in (\Gamma^{\mathrm{o}}_k \cup \{k\}) \setminus \{\bar{\gamma} \mid \ \}$ et pour tout $\gamma' \in R$, $Q_{\gamma'} \subseteq Q_B$. Pour tout $\gamma' \in R$, nous définissons un ensemble $R'_{\gamma} \subseteq Q_A \times Q_A$ égal à:

$$\{(p,q) \mid \exists (r,t,\gamma,T') \in Q_{\gamma}, p \xrightarrow{\gamma'} r, T_1 \in \Delta_A \text{ et } t \xrightarrow{\bar{\gamma}'} q, T_2 \in \Delta_A,\ T_1 \leq T \text{ et } T_2 \leq T'\}$$

La transition δ appartient à Δ_B si $(p,q) \in \left(\bigcup_{\gamma' \in R} R_{\gamma'}\right)^*$.

Fait 1. Pour tout $p,q \in Q_A$,toute instruction $\gamma \in \Gamma_k^{\mathrm{o}} \cup \{\, k \,\}$ et toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ avec $\mathrm{Last}(s) = \gamma$, s'il existe une exécution $\mathcal{E}_B = (T_B,C_B)$ de B commençant en s par l'état (p,q,γ,T) alors s appartient à $L_{p,q}$.

La preuve procède par récurrence sur la hauteur de T_B.

Fait 2. Pour tout $p,q \in Q_A$,tout $\gamma \in \Gamma_k^{\mathrm{o}} \cup \{\, k \,\}$ et toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ avec $\mathrm{Last}(s) = \gamma$, si s appartient à $L_{p,q}$ alors il existe une exécution $\mathcal{E}_B = (T_B,C_B)$ de B commençant en s par l'état (p,q,γ,T).

La preuve procède par récurrence sur la longueur de la boucle de A sur s en utilisant la décomposition du lemme 4.1.12.

Par conséquent et pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ avec $\mathrm{Last}(s) = \gamma_0$, $s \in \mathcal{S}(B)$ si et seulement si $s \in L_{p,q}$.

Par construction, B ne contient aucune occurrence des instructions $\bar{k}$ et $\bot_{k+1}$. Nous définissons l'automate C alternant sur Γ_k en remplaçant dans B toutes les occurrences de k par des ε. Par le lemme 4.1.3, il suit que pour tout $s = [\, s_1,\ldots,s_n \,]_{k+1} \in \mathrm{Stacks}_{k+1}(\Gamma)$ avec $\mathrm{Last}(s) = \gamma_0$, $s \in \mathcal{S}(B)$ si et seulement si $s_n \in \mathcal{S}(C)$. Ainsi l'automate C satisfait les conditions de l'énoncé.

Remarquons que bien que le nombre d'états de C soit polynomial en $|A|$, la taille de C est exponentielle en la taille de A. □

Exemple 4.3.47. Reprenons l'exemple 4.3.45. L'automate $A = (Q_A,I_A,F_A,\Delta_A)$ obtenu par ε-clôture de l'automate présenté dans la figure 4.1 est donné dans la figure 4.7.

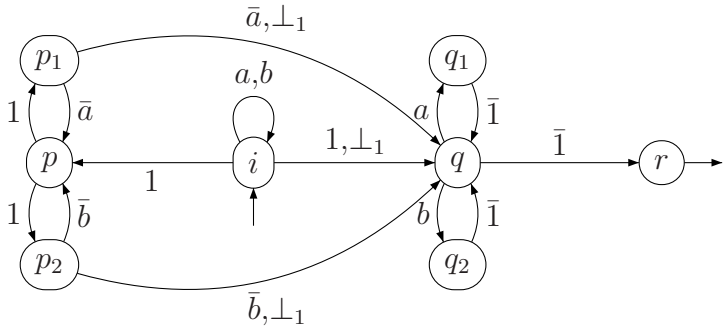


FIG. 4.7 – *L'ε-clôture de l'automate présenté dans la figure 4.1.*

Pour tout $\gamma \in \Gamma_1^{\mathrm{o}} \cup \{\, 1,\varepsilon \,\}$, nous construisons l'automate $A_{i,r}^{\gamma} = (Q,I,\Delta)$ alternant sur Γ_1 défini dans la preuve du lemme 4.3.46. L'ensemble des états

Q est $Q_A \times Q_A \times (\Gamma_1 \cup \{1\}) \times \mathrm{Sing}(\Gamma_1)$ et l'ensemble des états initiaux I est $\{(p,q,a,\emptyset), (p,q,a,\bot_1)\}$. Voici, l'ensemble des transitions de Δ «accessibles» depuis l'un des états initiaux.

$$
\begin{array}{lcl@{\qquad}lcl}
(i,r,\gamma,T),T & \to & ((q,q,1,\bot_1),\varepsilon) & (i,r,\gamma,T),T & \to & ((p,q,1,\emptyset),\varepsilon) \\
(p,q,\gamma',T),T & \to & ((p_1,q_1,1,\emptyset),\varepsilon) & (p,q,\gamma',T),T & \to & ((p_1,q_2,1,\emptyset),\varepsilon) \\
(p,q,\gamma',T),T & \to & ((p_2,q_1,1,\emptyset),\varepsilon) & (p,q,\gamma',T),T & \to & ((p_2,q_2,1,\emptyset),\varepsilon) \\
(p_1,q_1,1,\emptyset) & \to & (p,q,\bar{a},\emptyset),\bar{a}) & (p_1,q_1,1,\emptyset) & \to & (q,q,\bar{a},\bot_1),\bar{a}) \\
(p_2,q_2,1,\emptyset) & \to & (p,q,\bar{a},\emptyset),\bar{a}) & (p_2,q_2,1,\emptyset) & \to & (q,q,\bar{a},\bot_1),\bar{a}) \\
(q,q,\gamma',\bot_1),\bot_1 & \to & \emptyset & & & \\
\end{array}
$$

où $T \in \mathrm{Sing}(\Gamma_1^{\mathrm{T}})$ et $\gamma \in \{\bar{a},\bar{b},1\}$. Il est facile de vérifier que $\mathcal{S}(A^{\gamma}_{i,r})$ est égal à $\mathrm{Stacks}_1\Gamma.$ et donc $L_{i,r}$ est égal à $\mathrm{Stacks}_2(\Gamma)$.

En utilisant le lemme précédent, nous pouvons transformer tout automate sur Γ_{k+1} en un automate réduit sur Γ_{k+1} avec tests dans Alt_k.

Proposition 4.3.48. *Tout automate A sur Γ_{k+1} est équivalent à un automate B réduit sur Γ_k avec tests dans $\mathcal{L} \subseteq Alt_k$. De plus, $|\Delta_B|$ et $|\mathcal{L}|$ sont bornées par* $\exp[0](|Q_A|)$ *et chaque $L \in \mathcal{L}$ est accepté par un automate A_L alternant sur Γ_k avec $|Q_{A_L}|$ borné par* $\exp[0](|Q_A|)$.

Démonstration. Soit $A = (Q_A,I_A,F_A,\Delta_A)$ un automate sur Γ_{k+1}. Nous pouvons, sans perte de généralité, supposer que A ne contient pas d'ε-transitions. Nous construisons un automate $B = (Q_B,I_B,F_B,\mu_B,\Delta_B)$ réduit sur Γ_{k+1} avec tests dans un ensemble fini $\mathcal{L} \subset \mathrm{Alt}_k$.

L'ensemble $\mathcal{L}$ contient les langages acceptés par les automates $A^{\gamma}_{p,q}$ pour tout $p,q \in Q_A$ et $\gamma \in \Gamma_k^{\mathrm{o}} \cup \{k\}$. À ces langages, il faut ajouter les langages de tests associés aux états initiaux. Pour tout état $p \in Q_A$, nous noterons S_p le langage $\bigcup_{i \in I_A} \mathcal{S}(A^{\varepsilon}_{i,p})$. Par la proposition 4.3.16, le langage S_p appartient à Alt_k. Nous prenons donc $\mathcal{L}$ égal à:

$$
\begin{array}{rcl}
\mathcal{L} & = & \{\mathcal{S}(A^{\gamma}_{p,q}) \mid p,q \in Q_A \text{ et } \gamma \in \Gamma_k^{\mathrm{o}} \cup \{k\}\} \\
 & & \cup \{S_q \mid q \in Q_A\}
\end{array}
$$

où pour tout $p,q \in Q_A$ et $\gamma \in \Gamma^{\mathrm{o}}_{k+1}$, $A^{\gamma}_{p,q}$ est l'automate alternant sur Γ_k défini dans le lemme 4.3.46. Tout langage $L \in \mathcal{L}$ est accepté par un automate alternant sur Γ_k de taille polynomiale en $|A|$.

Nous pouvons maintenant définir B. L'ensemble des états Q_B est égal à $Q_A \times (\Gamma_k^{\mathrm{o}} \cup \{k,\varepsilon\})$. Les ensembles des états initiaux et finaux sont respectivement $I_B = Q_A \times \{\varepsilon\}$ et $F_B = F_A \times (\Gamma_k^{\mathrm{o}} \cup \{k,\varepsilon\})$. L'application μ_B est définie pour tout $(q,\varepsilon) \in Q_B$ par $\mu_B((q,\varepsilon)) = S_q$. L'ensemble des transitions Δ_B est donné

par:

$$\Delta_B = \{(p,\gamma') \xrightarrow{\gamma} (r,\gamma),T,T_{\mathcal{S}(A^{\gamma}_{q,r})} \mid \gamma' \in \Gamma^{\circ}_k \cup \{k,\varepsilon\}\gamma \in \Gamma^{\circ}_k \cup \{k\},\bar{\gamma} \neq \gamma' \text{ et } p \xrightarrow{\gamma} q,T \in \Delta_A\}.$$

Par construction, l'automate B est réduit (cf. remarque 4.3.23). Nous établissons maintenant que $\mathcal{S}(B) = \mathcal{S}(A)$. Par construction de B, nous avons $\mathcal{S}(B) \subseteq \mathcal{S}(A)$. Il reste à montrer l'inclusion réciproque.

Soit s une pile acceptée par A. Il existe un calcul $(p_0,s_0) \xrightarrow[A]{\gamma_1} \cdots \xrightarrow[A]{\gamma_n} (p_n,s_n)$ où $s_0 = [\,]_{k+1}$, $s_n = s$, $p_0 \in I_A$ et $p_n \in F_A$.

Notons ρ_s la suite réduite de s et pour tout $\ell \in [0,|\rho_s|]$, notons s_ℓ la pile de niveau $k+1$ ayant pour suite réduite $\rho(1)\dots\rho(\ell)$.

Il existe $i_1 < \ldots < i_{|\rho_s|} \in [0,n]$ tels que pour tout $\ell \in [1,|\rho_s|-1]$,

- $\gamma_{i_\ell+1} = \rho_s(\ell)$
- A boucle sur s_ℓ partant de $q_{i_\ell+1}$ et arrivant en $q_{i_{\ell+1}}$,
- A boucle sur $s_0 = [\,]_{k+1}$ partant de q_0 et arrivant en q_{i_0},
- A boucle sur $s_{|\rho_s|} = s$ partant de $q_{i_{|\rho_s|+1}}$ et arrivant en q_n

Il en résulte donc par construction de B et par définition des langages de $\mathcal{L}$ (cf. lemme 4.3.46) que B admet le calcul acceptant suivant:

$$((q_{i_1},\varepsilon),[\,]_{k+1}) \xrightarrow[B]{\rho_s(1)} ((q_{i_2},\rho(1)),s_1) \xrightarrow[B]{\rho_s(2)} \cdots \xrightarrow[B]{\rho_s(|\rho_s|)} ((q_n,\rho_s(|\rho_s|)),s).$$

Nous avons établi que $\mathcal{S}(A) \subseteq \mathcal{S}(B)$ et donc A et B acceptent le même langage.

Remarquons que comme la construction du lemme 4.3.46 prend un temps exponentiel en $|A|$, cette construction est elle aussi exponentielle. □

Il s'en suit maintenant par la proposition précédente et par le théorème 4.3.43, que tout automate A sur Γ_{k+1} est équivalent à un automate réduit sur Γ_{k+1} avec test dans Rat_k et que cette transformation peut être réalisée en temps $\exp[k](|A|)$. Pour un automate A alternant sur Γ_{k+1}, cette même normalisation est en temps $\exp[k+1](|Q_A|)$.

Proposition 4.3.49. *Tout automate A sur Γ_{k+1} est équivalent à un automate B réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset Rat_k$. De plus, $|B|$ est bornée par* $\exp[0](|A|)$ *et chaque $L \in \mathcal{L}$ est accepté par un automate A_L sur Γ_k de taille bornée par* $\exp[k](|A|)$.

Par le théorème 4.3.43, nous obtenons notre première étape de normalisation.

Théorème 4.3.50. *Les automates réduits sur Γ_{k+1} avec tests dans Rat_k acceptent précisément les langages de Rat_{k+1}.*

Démonstration. Les langages acceptés par les automates réduits sur Γ_{k+1} avec tests dans Rat_k appartiennent à Rat_{k+1} par la proposition 4.3.36.

Pour l'inclusion réciproque, soit $R \in \mathrm{Rat}_{k+1}$. Par la proposition 4.3.5, R est accepté par un automate A sur Γ_{k+1}. Par la proposition 4.3.49 et par le théorème 4.3.43, il s'en suit qu'il existe un automate réduit sur Γ_{k+1} avec tests dans Rat_k. □

Nous obtenons comme corollaire immédiat que pour tout ensemble rationnel de niveau $k+1$, l'ensemble des piles de niveau k apparaissant comme plus haute pile d'une pile de R est lui aussi rationnel.

Corollaire 4.3.51. *Pour tout $k \geq 1$ et pour tout langage R appartenant à $Rat_{k+1}(\Gamma)$, $\mathrm{top}_k(R)$ appartient à $Rat_k(\Gamma)$.*

Démonstration. Soit R un langage de $\mathrm{Rat}_{k+1}(\Gamma)$. Par le théorème 4.3.50, R est accepté par un automate A réduit sur Γ_{k+1} avec tests dans $\mathrm{Rat}_k(\Gamma)$. Par la remarque 4.3.7, A ne contient aucune occurrence des instructions $\perp_{k+1}$ ou $\overline{k}$. Considérons l'automate B sur Γ_k avec tests dans Rat_k obtenu en remplaçant dans A toutes les occurrences de l'instruction k par ε. Par le lemme 4.1.3, l'automate B accepte $\mathrm{top}_k(\mathcal{S}(A))$ et par le corollaire 4.3.44, $\mathcal{S}(B)$ appartient à $\mathrm{Rat}_k(\Gamma)$. □

Dans le paragraphe suivant, nous nous concentrons sur les automates réduits sur Γ_{k+1} avec tests dans Rat_k.
D'après la remarque 4.3.7, nous pouvons supposer que ces automates n'utilisent pas l'instruction $\perp_{k+1}$. De plus pour tout $\ell \in [1,k]$, l'opération de test de pile vide de niveau ℓ peut s'exprimer par un test rationnel de niveau k. En effet, $\mathcal{R}_{k+1}(\perp_\ell) = \mathrm{Test}^{k+1}_{S_\ell}$ où S_ℓ est l'ensemble des piles s de niveau k telles que $\mathrm{top}_\ell(s) = [\,]_\ell$. L'ensemble S_ℓ est égal à $\mathcal{R}_k((\Gamma_k^\circ)^* \cdot \perp_\ell)([\,]_k)$ et donc appartient à Rat_k.

Dans la suite, nous supposerons que les automates réduits que nous considérons n'utilisent pas les instructions de $\Gamma^{\mathrm{T}}_{k+1}$ et nous les omettrons dans la définition de ces automates.

4.3.4 Test du vide

Ce sous-paragraphe est dédié à l'étude de la complexité du test du vide des différents modèles d'automates introduits jusqu'à présent.

Pour les automates sur Γ_k, le test du vide peut se réduire en temps polynomial au test du vide du langage accepté par un automate à pile sur $\mathrm{COps}_k(\Gamma)$. En effet, par la proposition 4.2.2, le problème du test du vide des automates sur Γ_k est réductible en temps polynômial au problème du test du vide des automates à pile sur $\mathrm{Ops}_k(\Gamma)$. Par le théorème 4.1.23, ce problème est réductible en temps polynomial au problème du test du vide des automates à pile sur $\mathrm{COps}_k(\Gamma)$.

Dans [Eng91], l'auteur montre que le test du vide des automates sur $\mathrm{COps}_k(\Gamma)$ peut être réalisé en $\exp[k-1]$ (cf. théorème 4.1.23).

Voici un algorithme de test du vide des automates sur Γ_k et des automates alternants sur Γ_k qui nous permet en particulier de réobtenir le résultat de [Eng91]. D'après les propositions 4.3.48 et 4.3.41, les automates sur Γ_{k+1} et les automates alternants sur Γ_k peuvent tous être transformés en un automate équivalent réduit sur Γ_{k+1} avec tests dans Alt_k. Il nous suffit donc de nous concentrer sur les automates réduits sur Γ_k avec tests dans Alt_k.

Proposition 4.3.52. *Pour tout automate A réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset Alt_k$, il existe un automate B alternant sur Γ_k tel que $\mathrm{top}_k(\mathcal{S}(A)) = \mathcal{S}(B)$. De plus, si pour tout $L \in \mathcal{L}$, L est accepté par un automate A_L alternant sur Γ_k alors nous pouvons supposer que $|Q_B|$ est borné par $\exp[0](|A| + \sum_{L\in L} |Q_{A_L}|)$ et $|B|$ est bornée par $\exp[0](|A| + \sum_{L\in L} |A_L|)$.*

Démonstration. Soit $A = (Q_A, I_A, F_A, \mu_A, \Delta_A)$ un automate réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset \mathrm{Alt}_k$. Comme A est réduit, nous pouvons, sans perte de généralité, supposer que $\bar{k}$ n'étiquette aucune transition de A et que $\bot_{k+1}$ n'apparaît pas dans Δ_A (cf. remarque 4.3.7). Considérons l'automate B obtenu en remplaçant dans A toutes les occurrences de l'instruction k par ε. Par le lemme 4.1.3, il suit que $\mathcal{S}(B) = \mathrm{top}_k(\mathcal{S}(A))$. Par la proposition 4.3.34, il existe un automate C alternant élagué sur Γ_k avec test dans $\mathcal{L}$ équivalent à B. Finalement et par la proposition 4.3.36, il existe un automate D alternant sur Γ_k équivalent à C.
En combinant les complexités des différentes transformations mises en jeu, nous obtenons la borne annoncée. □

Nous pouvons donc, à l'aide de cette transformation, donner une borne supérieure sur la complexité du test du vide pour les différents modèles d'automates.

Proposition 4.3.53. *Les propositions suivantes sont vérifiées:*

1. *Le test du vide d'un automate A sur Γ_k est réalisable en temps $\exp[k-1](|A|)$.*
2. *Le test du vide d'un automate alternant sur Γ_k est réalisable en temps $\exp[k](|Q_A|)$.*

Démonstration. La preuve procède par récurrence sur le niveau k des automates.
Cas de base : $k = 1$. Le test du vide d'un automate sur Γ_1 se réduit en temps polynômial au test du vide d'un automate[8] réduit sur Γ_1. Ce test peut donc être effectué en temps polynomial.

Par la proposition 4.3.31, tout automate alternant sur Γ_1 peut être transformé en temps $\exp[1](|Q_A|)$ en un automate sur Γ_1 de taille au plus $\exp[1](|Q_A|)$. Le test

8. En d'autres termes, un automate fini étiqueté par Γ.

du vide d'un automate A alternant sur Γ_1 peut s'effectuer en temps $\exp[1](|Q_A|)$. Remarquons qu'il est important d'exprimer la complexité en fonction de $|Q_A|$ et non de $|A|$ car $|A|$ peut être exponentiellement plus grand que $|Q_A|$.
Etape de récurrence. Soit A un automate sur Γ_{k+1}. En combinant les propositions 4.3.48 et 4.3.52, nous pouvons construire un automate B alternant sur Γ_k tel que $\mathcal{S}(A)$ est vide si et seulement si $\mathcal{S}(B)$ est vide. De plus, $|Q_B|$ est bornée par $\exp[0](|A|)$ et $|B|$ peut être construit en $\exp[1](|A|)$. Par hypothèse de récurrence, le test du vide de B peut être réalisé en temps $\exp[k](|Q_B|)$. Le test du vide peut donc être réalisé en temps $\exp[k+1](|A|)$.
Remarquons que comme $k \geq 1$, la transformation de A en B qui est exponentielle est, en terme de complexité, masquée par le test du vide de B qui est au moins exponentiel.

Soit A un automate alternant sur Γ_{k+1}. En combinant le corollaire 4.3.42 et la proposition 4.3.52, nous pouvons construire un automate B alternant sur Γ_k tel que $\mathcal{S}(A)$ est vide si et seulement si $\mathcal{S}(B)$ est vide. De plus, $|Q_B|$ est bornée par $\exp[1](|Q_A|)$ et B peut être construit en temps $\exp[2](|Q_A|)$. Par hypothèse de récurrence, le test du vide de B peut être réalisé en temps $\exp[k](|Q_B|)$. Le test du vide de A peut donc s'effectuer en temps $\exp[k+1](|Q_A|)$.
Remarquons que comme $k \geq 1$, la transformation de A en B qui est doublement exponentielle par rapport au nombre d'états de A est, en terme de complexité, masquée par le test du vide B qui est lui aussi au moins doublement exponentiel en le nombre d'états de A. □

En utilisant la borne inférieure du test du vide des automates sur COps_k établie dans [Eng91] et rappelée dans le théorème 4.1.17, nous montrons que les complexités obtenues dans le théorème précédent sont minimales.

Théorème 4.3.54 ([Eng91])**.**

1. *Pour $k \geq 2$, il n'existe pas de procédure prenant en entrée un automate A sur Γ_k et décidant si $\mathcal{S}(A)$ est vide et s'exécutant en temps* $\exp[k-2](|A|)$.
2. *Pour $k \geq 1$, il n'existe pas de procédure prenant en entrée un automate A sur Γ_k et décidant si $\mathcal{S}(A)$ est vide et s'exécutant en temps* $\exp[k-1](|Q_A|)$.

Démonstration. Nous établissons les deux propriétés.

1. Raisonnons par l'absurde et supposons que pour un certain $k_0 \geq 2$, il existe un algorithme testant le vide des automates sur Γ_{k_0} travaillant en temps $\exp[k_0-2]$. Par le théorème 4.1.23 et la proposition 4.2.2, nous en déduisons un algorithme testant le vide du langage accepté par les automates sur COps_{k_0} travaillant en $\exp[k_0-2]$. Or par le théorème 4.1.17, le problème du test du vide des langages acceptés par les automates à piles sur COps_{k_0} est complet pour la classe de complexité $\bigcup_{d>1} \mathrm{DTIME}(2^{\uparrow^{k_0-1}(dn^2)})$. D'après

le théorème 2.19 de [HU79], $\exp[k_0-2](n) \subsetneq \bigcup_{d>1} \mathrm{DTIME}(2^{\uparrow^{k_0-1}(dn^2)})$: ce qui amène la contradiction.

2. Pour $k=1$, la propriété découle du fait que $|A|$ peut-être exponentielle en $|Q_A|$. Supposons par l'absurde que la propriété ne soit pas vérifiée. Pour un certain $k_0 \geq 2$, il existe donc une procédure testant le vide des automates alternants sur Γ_{k_0} et terminant en temps $\exp[k_0-1](|Q_A|)$. En reprenant la preuve de la proposition 4.3.53, nous pouvons construire pour tout automate A sur Γ_{k_0+1}, un automate B alternant sur Γ_{k_0} de taille polynomiale en $|A|$ en temps $\exp[1](|A|)$. Nous en déduisons donc une procédure décidant du vide des automates sur k_0+1 terminant en temps $\exp[k_0-1](|A|)$. Remarquons que comme $k_0 \geq 2$, k_0-1 est supérieur ou égal à 1 et la complexité de la procédure est bien en $\exp[k_0-1](|A|)$. Nous obtenons donc la contradiction avec la première partie de cette proposition.

□

Nous concluons ce paragraphe en montrant que le test de l'appartenance d'une pile au langage accepté par un automate sur Γ_k est aussi dur que le problème du test du vide des automates sur Γ_k. Formellement, le problème de l'appartenance pour les automates (resp. automates alternants) sur Γ_k est, étant donné en entrée une pile $s \in \mathrm{Stacks}_k(\Gamma)$ et un automate A (resp. automate alternant) sur Γ_k, de décider si s appartient à $\mathcal{S}(A)$.

Proposition 4.3.55. *Le problème de l'appartenance pour les automates (resp. automates alternants) sur Γ_k se réduit en temps polynomial au problème du test des automates (resp. automates alternants) sur Γ_k et vice-versa.*

Démonstration.

1. Considérons le cas des automates sur Γ_k. Pour la réduction directe, il suffit de remarquer que pour tout automate A sur Γ_k et pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, $\mathcal{R}(\mathcal{I}(A) \cdot \overline{\rho_s} \cdot \bot_k)([\,]_k)$ est vide si et seulement si s n'appartient pas à $\mathcal{S}(A)$. Pour la réciproque, il suffit de remarquer que $\mathcal{S}(A)$ n'est pas vide si et seulement si $[\,]_k \in \mathcal{R}(\mathcal{I}(A) \cdot (\Gamma_k^{\circ})^* \bot_k)$.
2. Considérons le cas des automates alternants sur Γ_k. Il suffit de remarquer que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ et pour tout automate A alternant sur Γ_k, $s \in \mathcal{S}(A)$ si et seulement si $\mathcal{S}(A) \cap \mathcal{S}(A_s) \neq \emptyset$ où A_s est un automate alternant sur Γ_k acceptant le singleton $\{\, s \,\}$. Remarquons que par la proposition 4.3.16, nous pouvons construire en temps polynomial un automate B alternant sur Γ_k et acceptant $\mathcal{S}(A) \cap \mathcal{S}(A_s)$. Pour la réduction réciproque, soit $A = (Q_A, I_A, \Delta_A)$ un automate alternant sur Γ_k. Nous allons construire un automate B tel que $\mathcal{S}(A)$ n'est pas vide si et seulement si $[\,]_k \in \mathcal{S}(B)$. L'automate B est égal à $(Q_A \cup \{\, \bullet \,\}, \{\, \bullet \,\}, \Delta_B)$ où $\bullet$ est un symbole n'appar-

tenant pas à Q_A et où l'ensemble des transitions Δ_B est défini par:

$$\Delta_B = \Delta_A \cup \{\bullet \to (\bullet,\gamma) \mid \gamma \in \Gamma_k\} \cup \{\bullet,\bot_k \to (\bullet,\varepsilon)\} \cup \{\bullet \to (i,\varepsilon) \mid i \in I_A\}.$$

□

4.4 Accepteurs finis déterministes

Au niveau 1, la fermeture par complémentaire de $\mathrm{Rat}_1(\Gamma)$ est obtenue en montrant que tout langage de Rat_1 est accepté par un automate déterministe et complet sur Γ [Kle56]. Dans ce paragraphe, nous montrons que la fermeture par complémentaire de $\mathrm{Rat}_{k+1}(\Gamma)$ pour tout $k \geq 1$ peut-être obtenue par un résultat similaire pour les automates réduits sur Γ_{k+1} avec tests dans $\mathrm{Rat}_k(\Gamma)$.

Dans ce but, nous définissons une notion naturelle de déterminisme et de complétude pour les automates réduits sur Γ_{k+1} avec tests dans $\mathrm{Rat}_k(\Gamma)$. Un tel automate $A = (Q_A,I_A,F_A,\mu(A),\Delta_A)$ est dit *déterministe* si quelque soit la configuration (p,s) de A et l'instruction $\gamma \in \Gamma_k^o \cup \{k\}$, il existe au plus un état q tel que $(p,s) \xrightarrow[A]{\gamma} (q,\mathcal{R}(\gamma)(s))$ et qu'il existe au plus un état initial $i \in I$ telle que $[\,]_{k+1} \in \mathrm{Dom}(\mu_A(i))$. L'automate A est dit *complet* si pour toute configuration (p,s) de A et pour toute $\gamma \in \Gamma_k^o \cup \{k\}$ telle que $\mathcal{R}(\gamma)(s)$ est définie et telle que $\bar{\gamma} \neq \mathrm{Last}(s)$, il existe au moins une transition $\delta \in \Delta_A$ étiquetée par γ applicable en (p,s). Remarquons que pour donner formellement la définition d'un automate réduit et complet, il est naturel de considérer des automates réduits de la forme décrite dans la remarque 4.3.7.

Définition 4.4.1. Un automate $A = (Q_A \times \Gamma_k^o \times \{k,\varepsilon\},I_A \times \{\varepsilon\},F_A,\mu_A,\Delta_A)$ réduit sur Γ_{k+1} avec tests dans un ensemble fini $\mathcal{L} \subset \mathrm{Rat}_k(\Gamma)$ est *déterministe* si pour tout $p,q,q' \in Q_A$ avec $q \neq q'$ et tout $\gamma \in \Gamma_k^o \cup \{k\}$ et $\gamma' \in \Gamma_k^o \cup \{k,\varepsilon\}$, on a

$$\begin{cases} \delta_1 = (p,\gamma') \xrightarrow{\gamma} (q,\gamma),T_1 \in \Delta_A \\ \delta_2 = (p,\gamma') \xrightarrow{\gamma} (q',\gamma),T_2 \in \Delta_A \end{cases} \quad \Rightarrow \quad \mathrm{Dom}(T_1) \cap \mathrm{Dom}(T_2) = \emptyset.$$

et de plus il existe au plus un $i \in I_A$ tel que $[\,]_{k+1} \in \mathrm{Dom}(\mu_A((i,\varepsilon)))$.
L'automate A est dit *complet* si pour tout $p \in Q_A$, $\gamma' \in \Gamma_k^o \cup \{k,\varepsilon\}$ et $\gamma \in \Gamma_k^o \cup \{k\}$ tel que $\bar{\gamma} \neq \gamma'$, l'union $\bigcup_{\delta=(p,\gamma') \xrightarrow{\gamma} (q,\gamma),T \in \Delta_A} \mathrm{Dom}(T)$ est égal à $\mathrm{Stacks}_k(\Gamma)$ et il existe au moins un état initial $i \in I_A$ tel que $[\,]_{k+1} \in \mathrm{Dom}(\mu((i,\varepsilon)))$.

Remarque 4.4.2. La notion d'automate déterministe et complet s'étend immédiatement aux automates sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$.

Dans la suite, nous dirons simplement automate déterministe et complet sur Γ_k avec tests dans $\mathcal{L} \subset \mathrm{Rat}_k$ au lieu d'automate réduit déterministe et complet sur Γ_k avec tests dans un ensemble fini $\mathcal{L} \subseteq \mathrm{Rat}_k(\Gamma)$.

La propriété clé de ces automates est que pour toute pile s de niveau k, il existe un et un seul calcul de l'automate partant de la pile vide de niveau k dans un état initial et arrivant à la pile s.

Lemme 4.4.3. *Pour tout automate A déterministe et complet sur Γ_{k+1} avec tests dans Rat_k et pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$, il existe un unique calcul de A partant de la pile vide $[\,]_{k+1}$ dans un état initial et arrivant en s. Nous noterons $A(s)$ l'état de A apparaissant dans la dernière configuration de ce calcul.*

Démonstration. Cette propriété est une récurrence immédiate sur la longueur de la suite réduite de la pile s. □

Une conséquence immédiate est que les langages acceptés par les automates déterministes et complets sur Γ_{k+1} avec tests dans Rat_k sont fermés par complémentaire. Pour tout automate $A = (Q_A,I_A,F_A,\mu_A,\Delta_A)$ déterministe et complet sur Γ_{k+1} avec tests dans $\mathcal{L} \subset \mathrm{Rat}_k$, le complémentaire de $\mathcal{S}(A)$ est accepté par l'automate $B = (Q_A,I_A,Q_A \setminus F_A,\mu_A,\Delta_A)$.

Proposition 4.4.4. *Les langages acceptés par les automates déterministes et complets sur Γ_{k+1} avec tests dans $\mathcal{L} \subset Rat_k$ sont clos par complémentaire.*

Nous présentons, dans le sous-paragraphe 4.4.1, une extension de la «méthode des sous-ensembles» qui permet de déterminiser et de compléter les automates réduits sur Γ_{k+1} avec tests dans Rat_k. Nous en déduisons la fermeture par complémentaire des ensembles de $\mathrm{Rat}_k(\Gamma)$. Dans le sous-paragraphe 4.4.2, nous présenterons une transformation permettant de passer d'un automate sur Γ_{k+1} à un automate déterministe et complet sur Γ_{k+1} avec test dans Rat_k dont la complexité est minimale.

4.4.1 Fermeture par complémentaire de $\mathrm{Rat}_k(\Gamma)$.

Nous montrons, par une adaptation de la «méthode des sous-ensembles», que tout automate réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset \mathrm{Rat}_k$ est équivalent à un automate déterministe et complet sur Γ_{k+1} avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$ où $\mathcal{L}^{\mathrm{c}}$ désigne l'ensemble des complémentaires des langages de $\mathcal{L}$ (*i.e.* $\mathcal{L}^{\mathrm{c}} = \{\mathrm{Stacks}_k(\Gamma) \setminus L \mid L \in \mathcal{L}\}$).

Proposition 4.4.5. *Tout automate A réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset Rat_k$ est équivalent à un automate B déterministe et complet sur Γ_{k+1} avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{C}}$. De plus, $|Q_B|$ est bornée par* $\exp[1](|Q_A|)$ *et $|B|$ est bornée par* $\exp[1](|A| + |\mathcal{L}|)$.

Démonstration. Soit $A = (Q_A \times (\Gamma_k^{\mathrm{o}} \cup \{\, k,\varepsilon \,\}),I_A \times \{\, \varepsilon \,\},F_A,\mu_A,\Delta_A)$ un automate réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset \mathrm{Rat}_k$. Nous construisons un automate $B =$

$(2^{Q_A} \times (\Gamma_k^{\circ} \cup \{\, k,\varepsilon \,\}),\{\, (I_0,\varepsilon) \,\},F_B,\Delta_B)$ déterministe et complet sur Γ_{k+1} avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$. L'ensemble $I_0 \subseteq I_A$ est défini comme $I_0 = \{i \mid [\,]_{k+1} \in \mathrm{Dom}(\mu_A(i))\}$, l'ensemble des états finaux F_B est égal à $\{(P,\gamma) \in 2^Q_A \times (\Gamma_k^{\circ} \cup \{\, k,\varepsilon \,\}) \mid P \cap F_A \neq \emptyset\}$. L'ensemble des transitions Δ_B est donné par:

$$\begin{aligned} \Delta_B \;=\; \{(P,\gamma') \xrightarrow{\gamma} (Q,\gamma)\,,T \mid\; & P \subseteq Q_A, \gamma' \in \Gamma_k^{\circ} \cup \{\, k,\varepsilon \,\}, \gamma \in \Gamma_k^{\circ} \cup \{\, k \,\}, \bar{\gamma} \neq \gamma', \\ & \exists \mathcal{L}' \subseteq \mathcal{L}, T = \{T_L^{k+1} \mid L \in \mathcal{L}'\} \cup \{T_{L^{\mathrm{c}}}^{k+1} \mid L \notin \mathcal{L}'\}, \\ & Q = \{q \in Q_A \mid \exists p \in P, p \xrightarrow{\gamma} q, T' \in \Delta_A, T' \subseteq T\}\}. \end{aligned}$$

Comme A est réduit, l'automate B l'est aussi. Montrons que B est déterministe et complet. Soient P,Q,Q' des états de Q_B avec $Q \neq Q'$ et soit $\gamma \in \Gamma_k^{\circ} \cup \{\, k \,\}$. Supposons que Δ_B contient deux transitions $P \xrightarrow{\gamma} Q,T_1$ et $P \xrightarrow{\gamma} Q',T_2$. Comme $Q \neq Q'$, il suit par définition de Δ_B que $T_1 \neq T_2$ et donc que $\mathrm{Dom}(T_1) \cap \mathrm{Dom}(T_2) = \emptyset$. Soient $P \in Q_B$ et $\gamma \in \Gamma_k^{\circ} \cup \{\, k \,\}$. L'union $\bigcup_{(P,\gamma') \xrightarrow{\gamma} (Q,\gamma), T \in \Delta_B} \mathrm{Dom}(T)$ est égale à:

$$\bigcup_{\mathcal{L}' \subseteq \mathcal{L}} \left(\bigcap_{L \in \mathcal{L}'} L \cap \bigcap_{L \in \mathcal{L} \setminus \mathcal{L}'} L^{\mathrm{c}} \right) = \mathrm{Stacks}_k(\Gamma).$$

La preuve de l'égalité entre $\mathcal{S}(A)$ et $\mathcal{S}(B)$ est une adaptation immédiate de la preuve de la «méthode des sous-ensembles».

Remarquons que cette construction fait intervenir le test d'appartenance aux langages de $\mathcal{L}$ (cf. proposition 4.3.55). □

Remarque 4.4.6. La construction présentée dans la preuve précédente reste valable pour les automates sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ (cf. remarque 4.4.2).

Exemple 4.4.7. La construction de la proposition 4.4.5 est illustrée dans la figure 4.8. En haut à droite, nous présentons un automate A réduit sur Γ_2 (où l'alphabet Γ est réduit au singleton $\{\, a \,\}$) avec tests dans $\mathcal{L} = \{\, E,F \,\} \subset \mathrm{Rat}_1$. En haut à gauche, nous donnons un automate B réduit sur Γ_2 avec tests dans $\mathcal{L}$ équivalent à A qui est dans la forme introduite par la remarque 4.3.7. Enfin en bas, nous donnons l'automate C déterministe et complet sur Γ_2 avec tests dans $\{\, E,F,E^{\mathrm{c}},F^{\mathrm{c}} \,\} \subset \mathrm{Rat}_1$ équivalent à B construit dans la proposition 4.4.5. Pour être concis, nous noterons, pour tout $\gamma \in \{\, \varepsilon,a,\bar{a},1 \,\}$, (γ,X_E) au lieu d'écrire l'ensemble $\{\, (\gamma,\{\, T_E,T_F \,\}),(\gamma,\{\, T_E,T_{F^{\mathrm{c}}} \,\}) \,\}$ et nous noterons simplement γ au lieu de l'ensemble $\{(\gamma,\{\, T_{L_1},T_{L_2} \,\}) \mid L_1 \in \{\, E,E^{\mathrm{c}} \,\}$ et $L_2 \in \{\, F,F^{\mathrm{c}} \,\}\}$.

Cette proposition permet de déduire la fermeture par complémentaire des ensembles de Rat_k.

Théorème 4.4.8. *Pour tout $k \geq 1$ et pour tout alphabet fini Γ, l'ensemble $Rat_k(\Gamma)$ est une algèbre de Boole.*

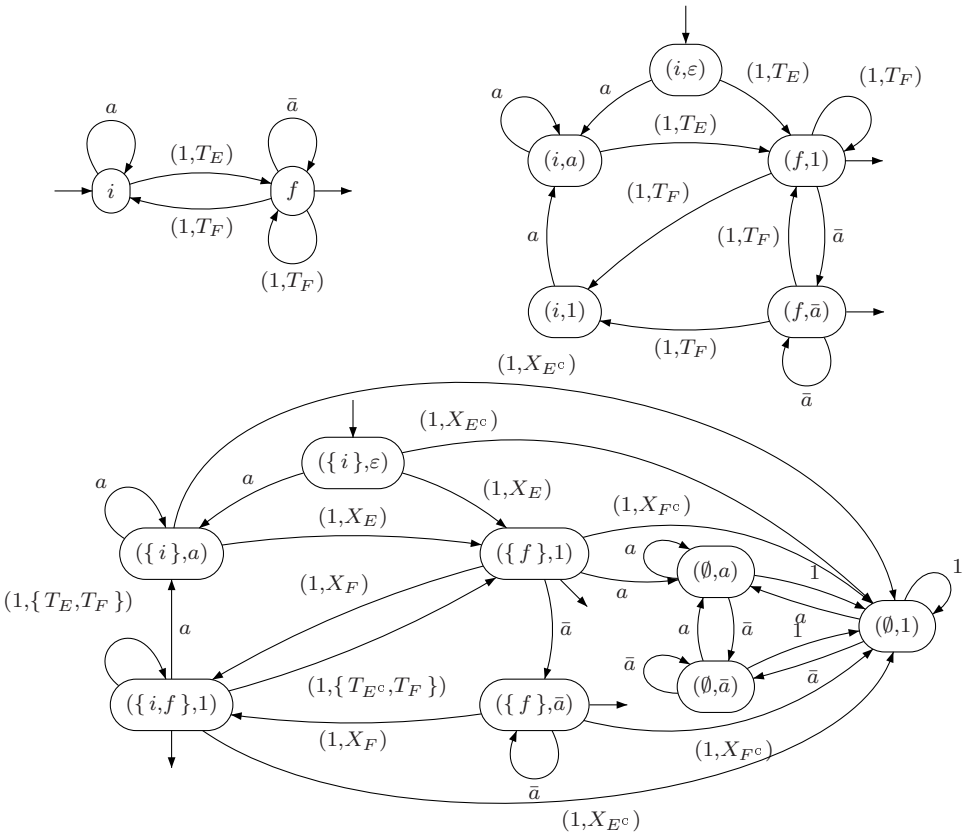


FIG. 4.8 – *Illustration de la proposition 4.4.5.*

Démonstration. Par définition, $\mathrm{Rat}_k(\Gamma)$ est clos par union et contient l'ensemble vide. Il suffit donc d'établir que $\mathrm{Rat}_k(\Gamma)$ est fermé par complémentaire. La preuve procède par récurrence sur le niveau k.

Cas de base : $k = 1$**.** Par le théorème 4.2.1, $\mathrm{Rat}_1(\Gamma) = \mathrm{Rat}(\Gamma^*)$ qui est une algèbre de Boole.

Etape de récurrence. Soit R dans $\mathrm{Rat}_{k+1}(\Gamma)$. D'après le théorème 4.3.50, il existe un automate A réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset \mathrm{Rat}_k$ acceptant R. Par la proposition 4.4.5, il existe un automate B déterministe et complet sur Γ_{k+1} avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$. Par hypothèse de récurrence, $\mathcal{L}^{\mathrm{c}}$ est inclus dans Rat_k. Par la proposition 4.4.4, le complémentaire de R, est accepté par un automate déterministe et complet sur Γ_{k+1} avec tests dans Rat_k. D'après le théorème 4.3.50, R^{c} appartient à $\mathrm{Rat}_{k+1}(\Gamma)$. □

Un corollaire immédiat du théorème précédent et de la proposition 4.3.49 est que tous les langages de $\mathrm{Rat}_{k+1}(\Gamma)$ sont acceptés par des automates déterministes

et complets sur Γ_{k+1} avec tests dans $\mathrm{Rat}_k(\Gamma)$.

Corollaire 4.4.9. *Pour tout $k \geq 1$, les automates déterministes et complets sur Γ_{k+1} avec tests dans $Rat_k(\Gamma)$ acceptent les langages de $Rat_{k+1}(\Gamma)$.*

Démonstration. Par la proposition 4.3.36, les langages des automates déterministes et complets sur Γ_{k+1} avec tests dans $\mathrm{Rat}_k(\Gamma)$ appartiennent à $\mathrm{Rat}_{k+1}(\Gamma)$. Pour l'inclusion inverse, considérons un langage R dans Rat_{k+1}. Par le théorème 4.3.50, R est accepté par un automate réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subseteq \mathrm{Rat}_k$. Par la proposition 4.4.5, R est accepté par un automate déterministe et complet sur Γ_{k+1} avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$. Par le théorème 4.4.8, $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}} \subset \mathrm{Rat}_k(\Gamma)$ et nous avons donc établi que R est accepté par un automate déterministe et complet sur Γ_{k+1} avec tests dans $\mathrm{Rat}_k(\Gamma)$. □

4.4.2 Complexité de la déterminisation

Nous commençons par analyser la complexité de la transformation d'un automate sur Γ_{k+1} en un automate déterministe et complet sur Γ_{k+1} avec tests dans Rat_k qui a été présentée dans le corollaire 4.4.9.

Nous introduisons pour cela la notion d'automate *entièrement déterministe et complet* sur Γ_k. Au niveau 1, un automate entièrement déterministe et complet sur Γ_1 est simplement un automate déterministe et complet sur Γ. Au niveau $k+1 \geq 2$, un automate entièrement déterministe et complet sur Γ_{k+1} est donné par un couple $(A,(A_i)_{i\in[1,n]})$ où $(A_i)_{i\in[1,n]}$ est une suite d'automates entièrement déterministes et complets sur Γ_k et où A est un automate réduit déterministe et complet sur Γ_{k+1} avec tests dans $\{\mathcal{S}(B_i) \mid i \in [1,n]\}$ dont la définition ne fait pas intervenir la fonction μ. Le langage accepté par l'automate est simplement le langage accepté par A et la taille de l'automate est la somme de la taille de A et de la taille des B_i.

Au niveau 1, la transformation d'un automate sur Γ_1 en un automate déterministe et complet sur Γ_1 est réalisable en temps $\exp[1](|A|)$ (cf. théorème 4.2.1 et la méthode des sous-ensembles). Pour un automate alternant sur Γ_1, la transformation en un automate déterministe et complet sur Γ_1 est doublement exponentielle en le nombre d'états de l'automate alternant sur Γ_1. Une première exponentielle vient de la transformation de l'automate en un automate réduit sur Γ (cf. proposition 4.3.31) et la deuxième vient de la déterminisation de l'automate sur Γ.

Au niveau $k+1 \geq 2$, la transformation d'un automate sur Γ_{k+1} en un automate entièrement déterministe et complet est réalisable en temps $\exp[k+1](|A|)$. Pour les automates alternants sur Γ_{k+1}, cette transformation peut s'effectuer en temps $\exp[k+2](|Q_A|)$.

Nous allons montrer que l'on peut diminuer toutes ces complexités d'un niveau

d'exponentiation[9] (cf. théorème 4.4.15) et que cette complexité est une borne inférieure pour la hauteur de la tour d'exponentielle (cf. proposition 4.4.19). Pour cela, nous transformons directement un automate alternant sur Γ_k en un automate entièrement déterministe et complet sur Γ_k.

4.4.2.1 Au niveau 1

Nous donnons une construction permettant de transformer un automate $A = (Q_A, I_A, \Delta_A)$ alternant sur Γ_1 en un automate déterministe et complet sur Γ en temps $\exp[1](|Q_A|)$.

Si pour toute pile $s \in \text{Stacks}_1(\Gamma)$, nous notons X_s l'ensemble des états $q \in Q_A$ tel qu'il existe un calcul de A partant de la pile s dans l'état q (*i.e.* $X_s = \{q \in Q_A \mid s \in \mathcal{S}_q(A)\}$). Il est aisé de vérifier que pour tout $\gamma \in \Gamma$ et pour toute pile $s \in \text{Stacks}_1(\Gamma)$, $X_{s\gamma}$ est entièrement déterminé par γ et X_s.
Nous pouvons alors définir un automate $B = (Q_B, I_B, F_B, \Delta_A)$ déterministe et complet en prenant $Q_B = 2^{Q_A}$, $I_B = \{ X_{[\,]_1} \}$ et $F_B = \{Q \subseteq Q_A \mid Q \cap I_A \neq \emptyset\}$. L'ensemble des transitions Δ_B est défini par:

$$\Delta_B = \{X_s \xrightarrow{\gamma} X_{s\gamma} \mid s \in \text{Stacks}_1(\Gamma) \text{ et } \gamma \in \Gamma\}.$$

Il vient immédiatement que B accepte le même langage de piles que A. Cette construction n'est cependant pas effective. La proposition 4.4.11 établit que l'automate B peut être construit à partir de A en temps $\exp[1](|A|)$. Le point clé de cette construction est donné par le lemme technique suivant.

Lemme 4.4.10. *Pour tout automate $A = (Q_A, I_A, \Delta_A)$ alternant sur Γ_1, tout $\gamma \in \Gamma$, tout $Q \subseteq Q_A$ et pour tout $q \in Q_A$, on a :*

- *soit pour tout $s \in \text{Stacks}_1(\Gamma)$, $X_s = Q$ implique $q \in X_{s\gamma}$,*
- *soit pour tout $s \in \text{Stacks}_1(\Gamma)$, $X_s = Q$ implique $q \notin X_{s\gamma}$.*

De plus, nous pouvons en décider en temps $\exp[1](|A|)$.

Démonstration. Soient $A = (Q_A, I_A, \Delta_A)$ alternant sur Γ_1, $Q_0 \subseteq Q_A$, $q_0 \in Q_A$ et $\gamma_0 \in \Gamma$. Nous allons adapter la construction de la preuve de la proposition 4.3.25 pour construire un automate B alternant réduit sur Γ_1 tel que pour toute pile $s \in \text{Stacks}_1(\Gamma)$ avec $X_s = Q_0$,

$$s\gamma_0 \in \mathcal{S}(B) \Leftrightarrow q_0 \in X_{s\gamma_0}.$$

Nous reprenons donc la construction de l'automate alternant réduit $B = (Q_B, I_B, \Delta_B)$ sur Γ_1 correspondant à l'automate A, présentée dans la proposition 4.3.25. La seule variation concerne l'ensemble des états initiaux I_B. Nous

9. Excepté bien entendu la complexité de la transformation d'un automate sur Γ_1 en un automate déterministe et complet sur Γ. Il est bien connu que la déterminisation des automates sur Γ est exponentielle dans le pire cas .

prenons:

$$I_B = \{(\gamma_0,f,\emptyset,R^{\downarrow}) \mid R^{\downarrow} \subset Q_A \times Q_A, q_0 \in \mathrm{Dom}(f) \text{ et } \xrightarrow[f]{\overline{\gamma_0}} (\mathrm{Dom}(f)) \subseteq Q_0\}.$$

Une adaptation immédiate de la preuve de la proposition 4.3.25 établit que B satisfait la propriété annoncée. Comme nous ne nous intéressons qu'au comportement de B sur des piles $s \in \mathrm{Stacks}_1(\Gamma)$ telles que $\mathrm{Last}(s) = \gamma_0$, nous pouvons supprimer toutes les transitions de B contenant une instruction dans $\overline{\Gamma}$ ou l'instruction $\perp_1$. Il s'en suit donc que $\mathcal{S}(B)$ est soit vide, ou soit égal à $\mathrm{Stacks}_1(\Gamma)$. De plus, nous pouvons tester le vide du langage accepté par B en $\mathcal{O}(|B|)$ [CDG$^+$]. □

En utilisant ce lemme, nous pouvons construire l'automate B en temps exponentiel par rapport au nombre d'états de A.

Proposition 4.4.11. *Pour tout automate A alternant sur Γ_1, il existe un automate B déterministe et complet sur Γ tel que $\mathcal{S}(A) = \mathcal{S}(B)$. De plus, l'automate B peut être construit en temps* $\exp[1](|Q_A|)$.

Démonstration. Par la proposition 4.3.55 et par la proposition 4.3.53, nous pouvons calculer $X_{[\,]_1}$ en temps $\exp[1](|Q_A|)$. Par le lemme 4.4.10, nous pouvons ,pour tout $Q \subseteq Q_A$ et pour tout $\gamma \in \Gamma$, calculer en temps $\exp[1](|Q_A|)$ l'ensemble Q_γ des états q tels que pour toute pile $s \in \mathrm{Stacks}_1(\Gamma)$, $X_s = Q$ implique $q \in X_{s\gamma}$. L'ensemble des transitions Δ_B est donc égal à

$$\Delta_B = \{(Q,\gamma,Q_\gamma) \mid Q \subseteq Q_A \text{ et } \gamma \in \Gamma\}.$$

L'automate B est constructible en $\exp[1](|Q_A|)$ et est par construction déterministe et complet. Par récurrence sur la longueur de la pile en utilisant le lemme 4.4.10, nous établissons que, pour toute pile $s \in \mathrm{Stacks}_1(\Gamma)$, il existe un calcul de B partant de la pile vide $[\,]_1$ dans l'état initial et arrivant s dans un état Q si et seulement si $X_s = Q$. Il en découle donc que $\mathcal{S}(A) = \mathcal{S}(B)$. □

4.4.2.2 Aux niveaux supérieurs

Nous allons adapter le principe de la preuve de la proposition 4.4.11 aux niveaux supérieurs.

Rappelons qu'au niveau 1, $X_{s\gamma}$ ne dépend que de X_s et de γ pour $s \in \mathrm{Stacks}_1(\Gamma)$ et $\gamma \in \Gamma$. Au niveau $k+1 \geq 2$, nous montrons, dans le lemme 4.4.12, que pour toute instruction $\gamma \in \Gamma_k^o \cup \{\,k\,\}$ et pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ telle que $\bar{\gamma} \neq \mathrm{Last}(s)$, si la pile $s' = \mathcal{R}(\gamma)(s)$ est définie alors $X_{s'}$ ne dépend que de de X_s, de γ et de $\mathrm{top}_k(s)$,. Cette dépendance est explicitée par le lemme technique suivant.

Lemme 4.4.12. *Pour tout automate A alternant sur Γ_{k+1}, tout $Q \subseteq Q_A$, pour tout $q \in Q_A$ et toute instruction $\gamma \in \Gamma_k^o \cup \{k\}$, il existe un automate $A^{\gamma}_{Q,q}$ alternant sur Γ_k tel que pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ avec $Last(s) \neq \bar{\gamma}$, $X_s = Q$ et $s' = \mathcal{R}(\gamma)(s)$ définie, nous avons:*

$$q \in X_{s'} \Leftrightarrow \mathrm{top}_k(s') \in \mathcal{S}(A^{\gamma}_{Q,q}).$$

De plus, $A^{\gamma}_{Q,q}$ peut être construit en temps $\exp[1](|Q_A|)$.

Démonstration. Soient $A = (Q_A, I_A, \Delta_A)$ un automate alternant sur Γ_{k+1}, $Q_0 \subseteq Q_A$, $q_0 \in Q_A$ et $\gamma_0 \in \Gamma_k \cup \{k\}$.

Dans un premier temps, nous adaptons la construction de la preuve de la proposition 4.3.25 pour construire un automate B alternant réduit sur Γ_{k+1} tel que pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ ayant $\mathrm{Last}(s) \neq \overline{\gamma_0}$, $X_s = Q_0$ et avec $s' = \mathcal{R}(\gamma)(s)$ définie, nous avons :

$$q_0 \in X_{s'} \Leftrightarrow s' \in \mathcal{S}(B).$$

Nous reprenons donc la construction de l'automate alternant réduit $B = (Q_B, I_B, \Delta_B)$ sur Γ_{k+1} correspondant à l'automate A présentée dans la proposition 4.3.25. La seule variation concerne l'ensemble des états initiaux I_B; nous prenons:

$$I_B = \{(\gamma_0, f, \emptyset, R^{\downarrow}) \mid R^{\downarrow} \subset Q_A \times Q_A, q_0 \in \mathrm{Dom}(f) \text{ et } \xrightarrow[f]{\overline{\gamma_0}} (\mathrm{Dom}(f)) \subseteq Q_0\}.$$

Une adaptation immédiate de la preuve de la proposition 4.3.25 établit que B satisfait la propriété annoncée. Comme nous ne nous intéressons qu'au comportement de B sur des piles $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ telles que $\mathrm{Last}(s) = \gamma_0$, nous pouvons supprimer toutes les transitions de B contenant les instructions $\overline{k}$ ou $\bot_{k+1}$.

L'automate $A^{\gamma_0}_{Q_0,q_0}$ alternant sur Γ_k est obtenu en remplaçant les occurrences de l'instruction k par ε dans B. Par la proposition 4.1.3, $\mathcal{S}(A^{\gamma_0}_{Q_0,q_0}) = \mathrm{top}_k(\mathcal{S}(B))$. L'automate $A^{\gamma_0}_{Q_0,q_0}$ satisfait donc la propriété annoncée. □

Le lemme précédent traite le cas des piles non-vides (*i.e.* $\mathrm{Last}(s) \neq \varepsilon$). Le lemme suivant couvre le cas de la pile vide $[\,]_{k+1}$ et est obtenu en combinant la proposition 4.3.25 et le lemme 4.3.40.

Lemme 4.4.13. *Pour tout automate A alternant sur Γ_{k+1} et pour tout $q \in Q_A$, il existe un automate A^{ε}_q alternant sur Γ_k tel que:*

$$[\,]_{k+1} \in \mathcal{S}_q(A) \quad \Leftrightarrow \quad [\,]_k \in \mathcal{S}(A^{\varepsilon}_q).$$

De plus, l'automate A^{ε}_q peut être construit en temps $\exp[1](|Q_A|)$.

En utilisant les deux lemmes précédents, nous pouvons construire, pour tout automate alternant sur Γ_{k+1}, un automate déterministe et complet avec tests dans $\mathrm{Alt}_k \cup \mathrm{Alt}_k^{\mathrm{c}}$ acceptant le même langage.

Proposition 4.4.14. *Pour tout $k \geq 1$ et tout automate A alternant sur Γ_{k+1}, il existe un automate B équivalent, déterministe et complet sur Γ_{k+1} avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$ où $\mathcal{L}$ est un ensemble fini d'éléments de $Alt_k(\Gamma)$.*
De plus, les tailles de B et de $\mathcal{L}$ sont bornées par $\exp[1](|Q_A|)$ *et chaque $L \in \mathcal{L}$ est accepté par un automate A_L alternant sur Γ_k de taille bornée par* $\exp[1](|Q_A|)$.
Tous ces éléments sont constructibles en temps $\exp[1](|Q_A|)$.

Démonstration. Soit $A = (Q_A,I_A,\Delta_A)$ un automate alternant sur Γ_{k+1}. Pour tout $Q \subseteq Q_A$, $q \in Q_A$ et $\gamma \in \Gamma_k^{\mathrm{o}} \cup \{ k \}$, nous noterons $A^{\gamma}_{Q,q}$ l'automate satisfaisant la propriété énoncée par le lemme 4.4.12 et nous noterons A^{ε}_q l'automate satisfaisant la propriété énoncée par le lemme 4.4.13.

Nous prenons $\mathcal{L} = \{\mathcal{S}(A^{\gamma}_{Q,q}) \mid Q \subseteq Q_A, q \in Q_A \text{ et } \gamma \in \Gamma_k^{\mathrm{o}} \cup \{ k \}\} \cup \{\mathcal{S}(A^{\varepsilon}_q) \mid q \in Q_A\}$. Nous définissons l'automate $B = (2^{Q_A} \times (\Gamma_k^{\mathrm{o}} \cup \{ k,\varepsilon \}), 2^{Q_A} \times \{ \varepsilon \}, F_B, \mu_B, \Delta_B)$ déterministe et complet avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$ acceptant $\mathcal{S}(A)$. L'ensemble des états finaux F_B est égal à $\{(P,\gamma) \subseteq 2^{Q_A} \times (\Gamma_k \cup \{ k,\varepsilon \}) \mid P \cap I_A \neq \emptyset\}$. L'application μ_B est définie pour tout $(Q,\varepsilon) \in I_B$ par $\mu((Q,\varepsilon)) = \{T^{k+1}_{\mathcal{S}(A^{\varepsilon}_q)} \mid q \in Q\} \cup \{T^{k+1}_{\mathcal{S}(A^{\varepsilon}_q)^{\mathrm{c}}} \mid q \notin Q\}$. Enfin, l'ensemble des transitions est défini par:

$$\{(P,\gamma') \xrightarrow{\gamma} (Q,\gamma),T \mid P,Q \subseteq Q_A, \gamma' \in \Gamma_k^{\mathrm{o}} \cup \{ k,\varepsilon \}, \gamma \in \Gamma_k^{\mathrm{o}} \cup \{ k \}, \gamma' \neq \bar{\gamma}, \\ T = \{T^{k+1}_{\mathcal{S}(A^{\gamma}_{P,q})} \mid q \in Q\} \cup \{T^{k+1}_{\mathcal{S}(A^{\gamma}_{P,q})^{\mathrm{c}}} \mid q \notin Q\}\}$$

Par construction, l'automate B est réduit, déterministe et complet sur Γ_{k+1}. En particulier, il existe un unique état $(I_0,\varepsilon) \in I_B$ tel que $[\,]_k \in \mathrm{Dom}(\mu_B((I_0,\varepsilon)))$ (*i.e.* $I_0 = \{q \in Q_A \mid [\,]_k \in \mathcal{S}(A^{\varepsilon}_q)\}$ par le lemme 4.4.13). Pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$, il existe un calcul de B partant de la pile vide $[\,]_k$ dans l'état (I_0,ε) et arrivant en s dans l'état (Q,γ) si et seulement si $X_s = Q$ et $\mathrm{Last}(s) = \gamma$. Cette propriété est établie par une récurrence immédiate sur la longueur de la suite réduite de s. Nous avons donc $\mathcal{S}(A) = \mathcal{S}(B)$. □

Le théorème ci-dessous donne une complexité améliorée pour la transformation des automates alternants et non-alternants sur Γ_k en des automates entièrement déterministes et complets sur Γ_k équivalents. Nous verrons dans la proposition 4.4.19 que cette complexité est minimale pour ce qui est de la hauteur de la tour d'exponentielles.

Théorème 4.4.15.

1. *Au niveau 1, pour tout automate A (resp. automate alternant) sur Γ_1, on peut construire un automate entièrement déterministe et complet équivalent à A en temps* $\exp[1]([A|)$ *(resp. en temps* $\exp[1](|Q_A|)$*).*

2. *Pour tout $k \geq 1$ et pour tout automate A (resp. automate alternant) sur Γ_{k+1}, on peut construire un automate B entièrement déterministe et complet sur Γ_{k+1} équivalent à A en temps* $\exp[k](|A|)$ *(resp. en temps* $\exp[k+1](|Q_A|)$*).*

Démonstration. Au niveau 1, la proposition 4.4.11 établit la propriété pour les automates alternants sur Γ_1. Comme par la proposition 4.3.14, il existe une transformation polynomiale des automates sur Γ_1 en des automates alternants sur Γ_1 équivalents, la propriété est aussi établie pour les automates sur Γ_1.

Nous allons établir, par récurrence sur le niveau k, l'existence d'une construction prenant un automate alternant sur Γ_k et donnant un automate équivalent entièrement déterministe et complet sur Γ_k et qui termine en temps $\exp[k](|Q_A|)$. Le cas de base a été établi dans la proposition 4.4.11. Passons à l'étape de récurrence. Soit $A = (Q_A, I_A, \Delta_A)$ un automate alternant sur Γ_{k+1}. Par la proposition 4.4.14, nous pouvons construire un automate équivalent B déterministe et complet avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$ avec $\mathcal{L} \subset \mathrm{Alt}_k(\Gamma)$ et où $|\mathcal{L}|$ est borné par $\exp[1](|Q_A|)$ et où chaque L est accepté par un automate A_L alternant sur Γ_k. Les automates B et A_L sont constructibles en temps $\exp[1](|Q_A|)$. Par hypothèse de récurrence, nous pouvons construire, pour chaque $L \in \mathcal{L}$, un automate B_L entièrement déterministe et complet sur Γ_k équivalent à A_L en temps $\exp[k](|A_L|)$. Par la proposition 4.4.4, nous pouvons construire en temps linéaire par rapport à la taille de B_L un automate, noté $B_{L^{\mathrm{c}}}$ entièrement déterministe et complet sur Γ_k acceptant le complémentaire du langage accepté par B_L.

L'automate A est donc équivalent à l'automate entièrement déterministe et complet $(B, (B_L)_{L\in\mathcal{L}} \cup (B_L^{\mathrm{c}})_{L\in\mathcal{L}})$ constructible en temps $\exp[k+1](|Q_A|)$.

Nous allons maintenant établir que, pour tout $k \geq 2$, il existe une construction prenant un automate sur Γ_{k+1} et donnant un automate équivalent entièrement déterministe et complet sur Γ_k et terminant en $\exp[k](|A|)$.

Soit A un automate sur Γ_{k+1} pour $k \geq 1$. En combinant le proposition 4.3.48 et la proposition 4.4.5, nous pouvons construire un automate B déterministe et complet avec tests dans $\mathcal{L} \cup \mathcal{L}^{\mathrm{c}}$ avec $\mathcal{L} \subset \mathrm{Alt}_k$ où la taille de $\mathcal{L}$ est bornée par $\exp[0](|A|)$ et où chaque $L \in \mathcal{L}$ est accepté par un automate A_L alternant sur Γ_k dont le nombre d'états est polynomial en la taille de A. De plus, les automates B et A_L peuvent être construits en temps $\exp[1](|A|)$.

Par ce qui précède, nous pouvons construire, pour chaque $L \in \mathcal{L}$, un automate B_L entièrement déterministe et complet sur Γ_k qui accepte le même langage que $\mathcal{S}(A_L)$ en temps $\exp[k](|A|)$. Par la proposition 4.4.4, nous pouvons également construire un automate $B_{L^{\mathrm{c}}}$ entièrement déterministe et complet sur Γ_k qui accepte le complémentaire de $\mathcal{S}(B_L)$ en temps linéaire par rapport à la taille B_L. Nous pouvons donc construire l'automate $(B, (B_L)_{L\in\mathcal{L}} \cup (B_{L^{\mathrm{c}}})_{L\in\mathcal{L}})$ entièrement déterministe et complet sur Γ_{k+1} qui accepte $\mathcal{S}(A)$ en temps $\exp[k](|A|)$. □

Remarque 4.4.16. Les automates entièrement déterministes et complets sur Γ_k correspondant aux automates A (resp. automates alternants B) sur Γ_k construits dans le théorème précédent sont tels que la taille de tous les ensembles de langages de tests apparaissant dans ces automates est bornée par $\exp[k-2](|A|)$ (resp. $\exp[k-1](|Q_B|)$).

Nous concluons ce sous-paragraphe en donnant la complexité des opérations booléennes sur les ensembles de $\mathrm{Rat}_k(\Gamma)$ quand ils sont représentés par des automates entièrement déterministes et complets sur Γ_k.

Proposition 4.4.17. *Pour tout automate A et B entièrement déterministes et complets sur Γ_k, les propositions suivantes sont satisfaites:*

1. *l'union des langages acceptés par A et B est acceptée par un automate C entièrement déterministe et complet de taille $|A| \cdot |B|$,*
2. *l'intersection des langages acceptés par A et B est acceptée par un automate C entièrement déterministe et complet de taille $|A| \cdot |B|$,*
3. *le complémentaire du langage accepté par A est accepté par un automate C entièrement déterministe et complet de taille $|A|$.*

Démonstration. La preuve procède par récurrence sur le niveau k des automates.
Cas $k = 1$. Ces propriétés sont bien connues (voir par exemple [HU79]).
Etape de récurrence. Soit $(A,(A_i)_{i\in[1,n]})$ et $(B,(B_i)_{i\in[1,m]})$ deux automates entièrement déterministes et complets sur Γ_{k+1}. Nous noterons $A = (Q_A,I_A,F_A,\mu_A,\Delta_A)$ et $B = (Q_B,I_B,F_B,\mu_B,\Delta_B)$. Nous allons définir un automate $C = (C,(A_i)_{i\in[1,n]} \cup (B_i)_{i\in[1,m]})$ entièrement déterministe et complet avec Γ_{k+1} acceptant $\mathcal{S}(A)\cap\mathcal{S}(B)$. Pour cela, nous définissons l'automate $C = (Q_C,I_C,F_C,\mu_C,\Delta_C)$ en prenant $Q_C = Q_A \times Q_B$, $I_C = I_A \times I_B$ et $F_C = F_A \times F_B$. L'application μ_C est définie pour tout $(i_A,i_B) \in I_C$ par $\mu_C((i_A,i_B)) = \mu_A(i_A) \cup \mu_B(i_B)$. Enfin, l'ensemble des transitions Δ_C est défini par:

$$\{(p_A,p_B) \xrightarrow{\gamma} (q_A,q_B),T_A \cup T_B \mid p_A \xrightarrow{\gamma} q_A,T_A \in \Delta_A \text{ et } p_B \xrightarrow{\gamma} q_B,T_B \in \Delta_B\}.$$

Il est aisé de montrer que comme A et B sont déterministes et complets, C l'est aussi. De plus par construction, C accepte $\mathcal{S}(A) \cap \mathcal{S}(B)$.

Pour l'union, il suffit de remplacer, dans la construction précédente, la définition de F_C par $F_C = (F_A \times Q_B) \cup (Q_A \times F_B)$.

Pour le complémentaire, il suffit comme nous l'avons déjà vu de complémenter l'ensemble des états finaux (cf. proposition 4.4.4). □

La proposition suivante étudie le problème de l'appartenance des automates entièrement déterministes et complets.

Proposition 4.4.18. *Le problème de l'appartenance pour les automates entièrement déterministes et complets sur Γ_{k+1} peut être résolu en temps polynomial.*

Démonstration. Avant d'établir la propriété, commençons par montrer que nous pouvons décider si une séquence d'instructions $\rho \in (\Gamma_k)^*$ est telle que $\mathcal{R}(\rho)([\,]_k)$ soit définie. Pour cela nous définissons la fonction partielle Suites_k de Γ_k^* dans $\Pi_{\ell\in[1,k]}\Gamma_\ell^*$. Cette fonction partielle est définie par récurrence sur la longueur de la suite ρ en prenant $\text{Suites}_k(\varepsilon) = (\varepsilon,\ldots,\varepsilon)$ et pour tout $\rho \in \Gamma_k^*$ et $\gamma \in \Gamma_k$ tels que $\text{Suites}_k(\rho)$ est définie et est égale à $(\rho'_1,\ldots,\rho'_k)$, nous définissons $\text{Suites}_k(\rho\gamma)$ par disjonction de cas sur γ:

Cas $\gamma = \bot_\ell \in \Gamma_k^{\text{T}}$**.** La fonction $\text{Suites}_k(\rho\gamma)$ est définie si $\rho'_\ell = \varepsilon$ et dans ce cas est égale à $\text{Suites}_k(\rho)$.

Cas $\gamma = \bar{\gamma}$**.** La fonction $\text{Suites}_k(\rho\gamma)$ est définie si $\rho'_1(|\rho'_1|) = \bar{\gamma}$ et dans ce cas est égale à $(\rho''_1,\rho'_2\gamma,\ldots,\rho'_k\gamma)$ où $\rho''_1 = \rho'_1(1)\cdots\rho'_1(|\rho'_1|-1)$.

Cas $\gamma = \Gamma$**.** La fonction $\text{Suites}_k(\rho\gamma)$ est définie et est égale à $(\rho'_1\gamma,\ldots,\rho'_k\gamma)$.

Cas $\gamma = \ell \in [1,k-1]$**.** La fonction $\text{Suites}_k(\rho\gamma)$ est définie. Elle est égale au uplet $(\rho_1,\ldots,\rho_k)$ où pour tout $i \le \ell$, $\rho_i = \rho'_i$ et pour tout $i > \ell$, $\rho_i = \rho'_i\ell$.

Cas $\gamma = \bar{\ell}$ **pour** $\ell \in [1,k-1]$**.** $\text{Suites}_k(\rho\gamma)$ est définie si $\rho'_{\ell+1}(|\rho'_{\ell+1}|) = \ell$. Dans ce cas, elle est égale à $(\rho_1,\ldots,\rho_k)$ où pour tout $i \le \ell$, $\rho_i = \rho'_i$, $\rho_{\ell+1} = \rho'_{\ell+1}(1),\ldots,\rho'_{\ell+1}(|\rho'_{\ell+1}|-1)$ et où pour tout $i > \ell$, $\rho_i = \rho'_i\gamma$.

Une récurrence immédiate sur la longueur de ρ établit que pour toute suite d'instructions $\rho \in \Gamma_k^*$, $\text{Suites}_k(\rho)$ est définie si et seulement si $\mathcal{R}(\rho)([\,]_k)$ l'est. De plus, si $s = \mathcal{R}(\rho)([\,]_k)$ alors $\text{Suites}_k(\rho)$ est égal à $(\rho_1,\ldots,\rho_k)$ où pour tout $\ell \in [1,k]$, ρ_ℓ est la suite réduite de $\text{top}_\ell(s)$. Il suit que nous pouvons décider en temps polynomial si une suite $\rho \in \Gamma_k^*$ est telle que $\mathcal{R}(\rho)([\,]_k)$ soit définie.

Nous allons maintenant démontrer la propriété annoncée. Pour cela, nous procédons encore une fois par récurrence sur le niveau k.

Cas $k = 1$**.** La propriété est triviale.

Etape de récurrence. Il nous faut construire un algorithme qui prenant en entrée une pile $s \in \text{Stacks}_{k+1}(\Gamma)$ (décrite par sa suite réduite d'instructions ρ_s) et un automate $(A,(A_i)_{i\in[1,n]})$ entièrement déterministe et complet sur Γ_{k+1} avec $A = (Q_A,I_A,F_A,\mu_A,\Delta_A)$ décide si $s \in \mathcal{S}(A)$. L'algorithme calcule par récurrence sur la longueur de la suite réduite ρ l'unique état q_ρ tel que A admette un calcul partant dans un état initial de la pile vide de niveau k et arrivant en $\mathcal{R}(\rho)([\,]_k)$ dans l'état q_ρ. Pour la suite vide ε, il suffit de décider pour tout $i \in [1,k]$ si $[\,]_k$ appartient à $\mathcal{S}(A_i)$. Par hypothèse de récurrence, nous pouvons ainsi déterminer l'unique état q_ε tel que $[\,]_k \in \mathcal{S}(A_i)$ pour tout $T^{k+1}_{\mathcal{S}(A_i)} \in \mu(q_\varepsilon)$. Supposons que nous avons calculé q_ρ, montrons comment calculer $q_\rho\gamma$ pour $\gamma \in \Gamma_k^{\text{o}}$. Nous commençons par calculer $\text{Suites}_k(\rho\gamma) = (\rho_1,\ldots,\rho_{k+1})$ qui est nécessairement définie car $\mathcal{R}(\rho\gamma)([\,]_k)$ est définie. L'état $q_{\rho\gamma}$ est l'unique état tel que $q_\rho \xrightarrow{\gamma} q_{\rho\gamma},T \in \Delta_A$ et tel que pour tout $T^{k+1}_{\mathcal{S}(A_i)} \in T$, la pile $\text{top}_k(\mathcal{R}(\rho\gamma)([\,]_k))$ de suite réduite ρ_k appartienne à $\mathcal{S}(A_i)$. Par hypothèse de récurrence, $q_{\rho\gamma}$ peut être calculé en temps polynomial. Nous pouvons donc décider en temps polynomial si $s \in \mathcal{S}(A)$. □

Une conséquence immédiate de la proposition précédente est que les complexités présentées dans le théorème 4.4.15 sont minimales en terme de la hauteur de la tour d'exponentielles. Au niveau 1, nous savons déjà que le plus petit automate déterministe et complet sur Γ équivalent à un automate sur Γ est dans le pire cas exponentiellement plus grand que l'automate de départ. Comme les automates sur Γ sont des cas particuliers d'automates sur Γ_1 et d'automates alternants sur Γ_1 (cf. proposition 4.3.14), il n'existe pas de transformation polynomiale des automates sur Γ_1 (resp. des automates alternants sur Γ_1) en des automates équivalents déterministes et complets sur Γ .

Proposition 4.4.19. *Pour tout $k \geq 2$, il n'existe pas de procédure transformant les automates (resp. automates alternants) sur Γ_k en des automates entièrement déterministes et complets sur Γ_k équivalents et terminant en temps* $\exp[k-2](|A|)$ *(resp.* $\exp[k-1](|Q_A|)$*).*

Démonstration. Commençons par établir cette borne inférieure pour les automates sur Γ_k. Supposons par l'absurde qu'il existe un niveau $k_0 \geq 2$ et une procédure transformant les automates (resp. automates alternants) sur Γ_k en des automates entièrement déterministes et complets sur Γ_k équivalents et terminant en temps $\exp[k-2](|A|)$. Par la proposition 4.4.18, il existerait une procédure résolvant le problème de l'appartenance pour les automates sur Γ_k terminant en temps $\exp[k-2]$. Par la proposition 4.3.55, ceci contredit le théorème 4.3.54.

La preuve pour les automates alternants sur Γ_k est une simple adaptation du cas précédent. □

Nous venons de voir que le test d'appartenance peut être réalisé en temps polynomial pour les automates entièrement déterministes et complets quelque soit leur niveau alors qu'il est $(k-1)$ fois exponentiel pour les automates sur Γ_k. Le test du vide, qui dans le cas des automates sur Γ_k est aussi dur (cf. proposition 4.3.55) que le problème de l'appartenance, reste coûteux pour les automates entièrement déterministes et complets sur Γ_k. En effet, comme le montre la propriété suivante, le test du vide des automates entièrement déterministes et complets sur Γ_k ne peut être réalisé en $\exp[k-4](|A|)$.

Proposition 4.4.20. *Pour $k \geq 4$, il n'existe pas de procédure résolvant le problème du test du vide des automates entièrement déterministes et complets sur Γ_k et terminant en temps* $\exp[k-4](|A|)$.

Démonstration. Pour établir cette borne inférieure, nous allons utiliser la borne inférieure sur la complexité du test du vide des automates sur Γ_k rappelée dans le théorème 4.3.54.

Nous allons construire pour tout automate A sur Γ_k, un automate B entièrement déterministe et complet sur Γ_{k+1} avec $|B| \leq 2^{|A|}$ tel que $\mathcal{S}(A)$ est vide si et seulement si $\mathcal{S}(B)$ l'est.

L'automate B est obtenu en déterminisant et en complétant l'automate A vu comme un automate sur le monoïde libre Γ_k et en insérant une instruction k avant chaque instruction de A. Ces insertions garantissent que l'automate sur Γ_{k+1} obtenu est bien réduit. Comme B ne possède pas de langages de tests, la déterminisation et la complétion dans le monoïde libre suffisent à garantir que l'automate B est entièrement déterministe et complet. Par la proposition 4.1.3, $\mathcal{S}(A) = \mathrm{top}_k(\mathcal{S}(B))$.

Supposons par l'absurde que pour un certain $k_0 \geq 4$, il existe une procédure décidant du vide des automates entièrement déterministes et complets sur Γ_k et terminant en temps $\exp[k_0-4](|A|)$ Nous pouvons donc en utilisant ce qui précède décider du vide des automate sur Γ_{k_0-1} en temps $\exp[k_0-4](\exp[1](|A|))$: ce qui amène la contradiction avec le théorème 4.3.54. □

La borne supérieure sur la complexité du vide de ces automates au niveau k est en $\exp[k-1](|A|)$. Elle est donnée par la proposition 4.3.52 et la proposition 4.3.53.

Avant de conclure ce sous-paragraphe, nous introduisons une forme normalisée des automates entièrement déterministes et complets sur Γ_k. Cette forme va nous permettre de simplifier les transformations présentées dans la suite et de nous permettre d'établir des résultats de complexité plus précis.

Définition 4.4.21. Un automate entièrement déterministe et complet sur Γ_k est dit *normalisé* si tout automate $(B,(B_i)_{i\in[1,n]})$ entièrement déterministe et complet sur Γ_ℓ, avec $1 < \ell \leq k$ et $B = (Q_B,I_B,F_B,\Delta_B)$, intervenant dans A est tel que pour tout $p,q \in Q_B$ et tout $\gamma \in \Gamma_{\ell-1} \cup \{\ell \mid \ \}$, il existe au plus une transition $p \xrightarrow{\gamma} q,T_{p,q}$ appartenant à Δ_B.

Tout automate entièrement déterministe et complet sur Γ_k est équivalent à un automate normalisé. Ceci découle immédiatement de la fermeture par union des Rat_ℓ pour $\ell \leq k$. Notons cependant que cette transformation est au moins exponentielle dans le pire cas.

Remarque 4.4.22. Pour $k \geq 1$, les automates entièrement déterministes et complets sur Γ_k correspondants aux automates alternants sur Γ_k construits dans le théorème 4.4.15 sont normalisés. Remarquons que les automates entièrement déterministes et complets sur Γ_k pour $k \geq 2$ construits dans le théorème 4.4.15 ne sont pas normalisés.

4.4.3 Automates sur Γ_k avec tests dans Rat_k

Nous concluons ce paragraphe en affinant la complexité de la transformation d'un automate A sur Γ_k avec tests dans $\mathcal{L} \subset \mathrm{Rat}_k(\Gamma)$ en un automate B entièrement déterministe et complet sur Γ_k. Nous considérons le cas où chaque

langage $L \in \mathcal{L}$ est accepté par un automate A_L entièrement déterministe et complet normalisé (cf. définition 4.4.21). Les résultats obtenus sont résumés dans le théorème 4.4.25. Cette section ne présente pas d'intérêt particulier en dehors de l'établissement de ce théorème.

Par la proposition 4.3.38 et par le théorème 4.4.15, nous obtenons un automate B de taille bornée par $\exp[k](|A| + \sum_{L\in\mathcal{L}} |A_L|)$ pour $k \geq 1$. Pour $k \geq 2$, nous affinons cette complexité et nous obtenons une borne en $\exp[k-1](|A| + \prod_{L\in\mathcal{L}} |A_L|)$.

Avant de présenter les différentes étapes de cette construction, nous introduisons les notions communes à toutes ces étapes.
Soit $A = (Q_A, I_A, F_A, \Delta_A)$ un automate sur Γ_k avec tests dans $\mathcal{L} = \{ L_1, \ldots, L_n \} \subset \mathrm{Rat}_k(\Gamma)$. Supposons de plus que pour tout $\ell \in [1,n]$, L_ℓ est accepté par un automate $A_\ell = (B_\ell, (C_i^\ell)_{i\in[1,n_\ell]})$ entièrement déterministe et complet normalisé avec $B_\ell = (Q_\ell, \{ i_\ell \}, F_\ell, \Delta_\ell)$. Pour $k \geq 2$, nous noterons $\mathcal{L}' \subset \mathrm{Rat}_{k-1}(\Gamma)$ l'ensemble $\{\mathcal{S}(C_i^\ell) \mid \ell \in [1,n] \text{ et } i \in [1,n_\ell]\}$.

Pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$, nous notons X_s le uplet $(A_1(s), \ldots, A_n(s)) \in \prod_{\ell\in[1,n]} Q_\ell$ (cf. lemme 4.4.3).
Il est aisé de vérifier que $X_{[\,]_k}$ est égal à $(i_1, \ldots, i_n)$ et que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ avec $X_s = (q_1, \ldots, q_n)$, la pile s appartient à L_ℓ si et seulement si q_ℓ appartient à F_ℓ.
Pour $k \geq 2$, la propriété clé est que pour tout uplet $e = (e_1, \ldots, e_n)$ et $f = (f_1, \ldots, f_n)$ dans $\prod_{\ell\in[1,n]} Q_\ell$ et pour tout $\gamma \in \Gamma_{k-1} \cup \{ k \}$, il existe un sous-ensemble $R_{e,f}^\gamma$ de $\mathcal{L}'$ tel que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ avec $\mathrm{Last}(s) \neq \bar{\gamma}$, $X_s = e$ et telle que $s' = \mathcal{R}(\gamma)(s)$ soit définie,

$$X_{s'} = f \Leftrightarrow \mathrm{top}_{k-1}(s') \in \bigcap_{R\in R_{e,f}^\gamma} R.$$

L'ensemble $R_{e,f}^\gamma$ est défini comme suit:

- $R_{e,f}^\gamma = \{ \emptyset \}$ s'il existe $\ell \in [1,n]$ tel qu'il n'existe pas de transition de la forme $p_\ell \xrightarrow{\gamma} q_\ell, T$ dans Δ_ℓ.
- $R_{e,f}^\gamma = \bigcup_{\ell\in[1,n]} T_\ell$ si pour tout $\ell \in [1,n]$, il existe une transition $p_\ell \xrightarrow{\gamma} q_\ell, T_\ell$ dans Δ_ℓ. Notons que comme l' automate A_ℓ est normalisé, cette transition est unique.

La première étape reprend l'approche du sous-paragraphe 4.3.3.2 et transforme un automate sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ en un automate équivalent réduit sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$.

Proposition 4.4.23. *Pour tout automate A sur Γ_k avec tests dans $\mathcal{L} \subset Rat_k(\Gamma)$, il existe un automate réduit B sur Γ_k avec tests dans $\mathcal{L}' \subset Rat_k(\Gamma)$ acceptant $\mathcal{S}(A)$. De plus, si chaque $L \in \mathcal{L}$ est accepté par un automate A_L entièrement déterministe et complet normalisé, $|Q_B|$ est bornée par* $\exp[0](|A|)$, *$|B|$ et $|\mathcal{L}'|$ sont*

bornées par $\exp[0](|A| + \prod_{L\in\mathcal{L}} |A_L|)$ *et chaque* $L \in \mathcal{L}'$ *est accepté par un automate* B_L *entièrement déterministe et complet normalisé de taille bornée par* $\exp[k-1](|A| + \prod_{L\in\mathcal{L}} |A_L|)$ *si* $k \geq 2$ *et* $\exp[1](|A| + \prod_{L\in\mathcal{L}} |A_L|)$ *si* $k = 1$.

Démonstration. Nous considérons le cas $k \geq 2$. Le cas $k = 1$ est une adaptation immédiate. Soit $A = (Q_A,I_A,F_A,\Delta_A)$ un automate sur Γ_k avec tests dans $\mathcal{L} = \{ L_1,\ldots,L_n \} \subset \mathrm{Rat}_k(\Gamma)$. Nous reprenons les notations présentées au début de ce sous-paragraphe. Pour tout $p,q \in Q_A$, nous reprenons aussi la définition des langages de boucles $L_{p,q}$ introduits dans le sous-paragraphe 4.3.3.2.

Comme l'établit le lemme 4.3.46, dans le cas où $\mathcal{L} = \emptyset$, l'appartenance d'une pile $s \in \mathrm{Stacks}_k(\Gamma)$ ne dépend que de $\mathrm{Last}(s)$ et $\mathrm{top}_{k-1}(s)$. Par une adaptation immédiate de la preuve du lemme 4.3.46, nous établissons que lorsque $\mathcal{L}$ n'est pas vide, l'appartenance d'une pile $s \in \mathrm{Stacks}_k(\Gamma)$ ne dépend que de $\mathrm{Last}(s)$, $\mathrm{top}_{k-1}(s)$ et de X_s.

Plus précisément, pour tout $p,q \in Q_A$ et $\gamma \in \Gamma^{\circ}_{k-1} \cup \{ k,\varepsilon \}$ et tout $d \in \prod_{\ell\in[1,n]} Q_\ell$, il existe un automate $A^{\gamma,d}_{p,q}$ alternant sur Γ_{k-1} avec tests dans $\mathcal{L}'$ tel que pour toute pile $s \in \mathrm{Stacks}_k(\Gamma)$ avec $\mathrm{Last}(s) = \gamma$ et $X_s = d$, $s \in L_{p,q}$ si et seulement si $\mathrm{top}_{k-1}(s) \in \mathcal{S}(A^{\gamma,d}_{p,q})$.

L'ensemble des états de $A^{\gamma,d}_{p,q}$ est égal à $(\Gamma^{\circ}_{k-1} \cup \{ k,\varepsilon \}) \times Q_A \times (\prod_{\ell\in[1,n]} |Q_\ell|)$. La construction de l'automate est une adaptation immédiate de la construction du lemme 4.3.46.

Le nombre d'états de cet automate est borné par $\exp[0](|A| + \prod_{\ell\in[1,n]} |Q_\ell|)$. Par la proposition 4.3.41, $A^{\gamma,d}_{p,q}$ est équivalent à un automate alternant sur Γ_{k-1} sans tests dont le nombre d'états est borné par $\exp[0](|A| + \prod_{\ell\in[1,n]} |Q_\ell|)$.

Par le théorème 4.4.15, il existe un automate $B^{\gamma,d}_{p,q}$ entièrement déterministe et complet sur Γ_{k-1} de taille bornée par $\exp[k-1](|A| + \prod_{\ell\in[1,n]} Q_\ell)$ et équivalent à $A^{\gamma,d}_{p,q}$. Nous considérons $B^{\gamma,d}_{p,q}$ comme un automate entièrement déterministe et complet sur Γ_k.

En adaptant la preuve de la proposition 4.3.41, nous construisons un automate $D = (Q_D,I_D,F_D,\Delta_D)$ réduit sur Γ_k avec tests dans:

$$\begin{array}{rcl} \mathcal{L}' & = & \{\mathcal{S}(A^q_\ell) \mid \ell \in [1,n] \text{ et } q \in Q_\ell\} \\ & \cup & \{\mathcal{S}(A^{\gamma,d}_{p,q}) \mid p,q \in Q_A, \gamma \in \Gamma^{\circ}_{k-1} \cup \{ k,\varepsilon \} \text{ et } d \in \prod_{\ell\in[1,n]} |Q_\ell|\}. \end{array}$$

où pour tout $\ell \in [1,n]$ et pour tout $q \in Q_\ell$, A^q_ℓ désigne l'automate entièrement déterministe et complet sur Γ_k égal à $((Q_\ell,\{ i_\ell \},\{ q \},\Delta_\ell),(C^\ell_i)_{i\in[1,n_\ell]})$. Tous les langages de $\mathcal{L}'$ sont bien acceptés par des automates déterministes et complets sur Γ_k de taille bornée par $\exp[k-1](|A| + \prod_{\ell\in[1,n]} |Q_\ell|)$.

L'ensemble des états Q_D est égal à $Q_A \times (\Gamma^{\circ}_{k-1} \cup \{ k,\varepsilon \})$. L'ensemble I_D des états initiaux est égal à $\{(p,\varepsilon) \in Q_A \times \{ \varepsilon \} \mid i \in I_A, [\,]_{k-1} \in \mathcal{S}(A^{\varepsilon,X_{[\,]_k}}_{i,p})\}$. Par la proposition 4.4.18, l'ensemble I_D peut être calculé en temps $\exp[k-1](|A| +$

$\prod_{\ell\in[1,n]}|Q_\ell|$). L'ensemble F_D des états finaux est $F_D = F_A \times (\Gamma^{\circ}_{k-1} \cup \{\, k,\varepsilon \,\})$. L'ensemble des transitions Δ_B est donné par:

$$\begin{array}{lll} \{(p,\gamma') \xrightarrow{\gamma} (r,\gamma),T' & | & \gamma' \in \Gamma^{\circ}_{k-1} \cup \{\, k,\varepsilon \,\}, \gamma \in \Gamma^{\circ}_{k-1} \cup \{\, k \,\}, \bar{\gamma} \neq \gamma' \\ & & p \xrightarrow{\gamma} q, T \in \Delta_A, d = (q_1,\ldots,q_n) \in \prod_{\ell\in[1,n]}|Q_\ell| \\ & & \text{pour tout } i \in [1,n],\ T_{L_i} \in T \Rightarrow q_i \in Q_i, \\ & & \text{et } T' = \{T_{\mathcal{S}(A^{q_\ell}_\ell)} \mid \ell \in [1,n]\} \cup \{\, T_{\mathcal{S}(A^{\gamma,d}_{q,r})} \,\} \qquad \}. \end{array}$$

Par construction, l'automate D est réduit (cf. remarque 4.3.23) et en adaptant la preuve de la proposition 4.3.49, nous montrons que D accepte $\mathcal{S}(A)$.

□

La deuxième partie de la preuve est plus intéressante et consiste à remplacer les tests de niveau k dans un automate réduit sur Γ_k avec tests $\mathcal{L} \subset \mathrm{Rat}_k(\Gamma)$ par des tests de niveau $k-1$.

Proposition 4.4.24. *Pour tout automate A réduit sur Γ_k avec tests dans $\mathcal{L} \subset Rat_k(\Gamma)$, il existe un automate B réduit Γ_k avec tests dans $\mathcal{L}' \subset Rat_{k-1}(\Gamma)$ acceptant le même langage.*
De plus, si chaque $L \in \mathcal{L}$ est accepté par un automate A_L entièrement déterministe et complet normalisé, $|B|$ est bornée par $\exp[0](|A| + \prod_{L\in\mathcal{L}}|A_L|)$ et $|\mathcal{L}'|$ est bornée par $\exp[0](|A| + \sum_{L\in\mathcal{L}}|A_L|)$ et chaque $L \in \mathcal{L}'$ est accepté par un automate B_L entièrement déterministe et complet normalisé de taille bornée par $\exp[0](|A| + \sum_{L\in\mathcal{L}}|A_L|)$.
Enfin, si A est déterministe et complet alors B l'est aussi.

Démonstration. Considérons le cas $k \geq 2$. Le cas $k = 1$ est une adaptation immédiate. Soit $A = (Q_A,I_A,F_A,\Delta_A)$ un automate sur Γ_k avec tests dans $\mathcal{L} = \{\, L_1,\ldots,L_n \,\} \subset \mathrm{Rat}_k(\Gamma)$. Nous reprenons les notations introduites au début de ce sous-paragraphe.

Nous pouvons maintenant définir un automate $D = (Q_D,I_D,F_D,\Delta_D)$ réduit sur Γ_k avec tests dans $\mathcal{L}'$ équivalent à A. L'ensemble des états Q_D est égal à $Q_A \times (\prod_{\ell\in[1,n]} Q_\ell)$. L'ensemble des états initiaux et finaux sont respectivement $I_A \times X_{[\,]_k}$ et $Q_A \times (\prod_{\ell\in[1,n]} Q_\ell)$. L'ensemble Δ_D des transitions est défini par :

$$\begin{array}{r} \{(p,e) \xrightarrow{\gamma} (q,f), \mathcal{T}_{R^{\gamma}_{e,f}} \mid p \xrightarrow{\gamma} q, \{\, T_{L_{i_1}},\ldots,T_{L_{im}} \,\} \in \Delta_A, e,f \in \prod_{\ell\in[1,n]} Q_\ell \\ \text{et pour tout } j \in [1,m], f_{i_j} \in F_j\}. \end{array}$$

Par construction, l'automate D est réduit et accepte $\mathcal{S}(A)$.

Supposons maintenant que A soit déterministe et complet (cf. remarque 4.4.2). Montrons que D est déterministe et complet. Comme A est déterministe, pour tout état $p \in Q_A$ et pour tout e et $f \in \prod_{\ell\in[1,n]} Q_\ell$, il existe au plus un état

$q \in Q_A$ tel que $(p,e) \xrightarrow{\gamma} (q,f), \mathcal{T}_{R^\gamma_{e,f}}$ appartienne à Δ_D. Il suffit alors de remarquer que pour tout γ et pour tout e,f et f' dans $\prod_{\ell\in[1,n]} Q_\ell$ avec $f \neq f'$, nous avons $\bigcap_{R\in R^\gamma_{e,f}} R \cap \bigcap_{R\in R^\gamma_{e,f'}} = \emptyset$ car les automates A_L sont déterministes. La complétude de D se dérive de la complétude de A et des automates A_L. □

En combinant les deux propositions précédentes et la proposition 4.4.5, nous transformons les automates sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ en des automates équivalents entièrement déterministes et complets sur Γ_k.

Théorème 4.4.25.

1. *Pour tout automate A sur Γ_1 avec tests dans $\mathcal{L} \subset Rat_1(\Gamma)$, il existe une automate B entièrement déterministe et complet sur Γ_1 acceptant $\mathcal{S}(A)$. De plus si chaque $L \in \mathcal{L}$ est accepté par un automate entièrement déterministe et complet, la taille de B est bornée par* $\exp[1](|A| + \sum_{L\in\mathcal{L}} |A_L|)$.
2. *Pour $k \geq 2$ et pour tout automate A sur Γ_k avec tests dans $\mathcal{L} \subset Rat_k(\Gamma)$, il existe une automate B entièrement déterministe et complet sur Γ_k acceptant $\mathcal{S}(A)$. De plus si chaque $L \in \mathcal{L}$ est accepté par un automate entièrement déterministe et complet normalisé, la taille de B est bornée par* $\exp[k-1](|A| + \prod_{L\in\mathcal{L}} |A_L|)$.

Démonstration. Comme nous l'avons vu, le premier point découle de la proposition 4.3.38 et du théorème 4.4.15.

Établissons le second point. Soit $k \geq 2$ et A un automate sur Γ_k avec tests dans $\mathcal{L} \subset \mathrm{Rat}_k(\Gamma)$. Pour tout $L \in \mathcal{L}$, soit A_L un automate entièrement déterministe et complet normalisé acceptant L. Par la proposition 4.4.23, il existe un automate B réduit sur Γ_k avec tests $\mathcal{L}' \subset \mathrm{Rat}_k(\Gamma)$ acceptant $\mathcal{S}(A)$. La taille de Q_B est bornée par $\exp[0](|A|)$, $|B|$ et $|\mathcal{L}'|$ sont bornées par $\exp[0](|A| + \prod_{L\in\mathcal{L}} |A_L|)$ et chaque $L \in \mathcal{L}'$ est accepté par un automate B_L entièrement déterministe et complet normalisé de taille bornée par $\exp[k-1](|A|+\prod_{L\in\mathcal{L}} |A_L|)$. Par la proposition 4.4.5 et la remarque 4.4.6, nous construisons un automate C déterministe et complet sur Γ_k avec tests dans $\mathcal{L}' \cup (\mathcal{L}')^{\mathrm{c}} \subset \mathrm{Rat}_k$ acceptant $\mathcal{S}(B)$ dont la taille est bornée par $\exp[1](|A| + \prod_{L\in\mathcal{L}} |A_L|)$. Par la proposition 4.4.24, il existe un automate D réduit déterministe et complet sur Γ_k avec tests $\mathcal{L}'' \subset \mathrm{Rat}_{k-1}(\Gamma)$ acceptant $\mathcal{S}(C)$. De plus, nous pouvons supposer que $|D|$ est bornée par $\exp[0](|B| + \prod_{L\in\mathcal{L}'} |B_L|)$ et $|\mathcal{L}'|$ est bornée par $\exp[0](|B| + \sum_{L\in\mathcal{L}'} |B_L|)$ et chaque $L \in \mathcal{L}''$ est accepté par un automate C_L entièrement déterministe et complet normalisé de taille bornée par $\exp[0](|B| + \sum_{L\in\mathcal{L}} |B_L|)$. L'automate entièrement déterministe et complet recherché est donc $(D,(C_L)_{L\in\mathcal{L}''})$ dont la taille est bornée par $\exp[k-1](|A| + \prod_{L\in\mathcal{L}} |A_L|)$ car $k - 1 \geq 1$.

□

4.5 Relations préfixe-reconnaissables d'ordre supérieur

Ce paragraphe étudie les relations sur les piles de niveau k induites par les ensembles rationnels de suites d'instructions dans $(\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k(\Gamma)})^*$. Ces relations apparaissent naturellement dans l'étude des langages de $\mathrm{Rat}_{k+1}(\Gamma)$. En effet, en reformulant le théorème 4.3.50, il suit que tout ensemble de $\mathrm{Rat}_{k+1}(\Gamma)$ est égal à une union finie d'ensembles dans:

$$\mathcal{R}\left(\mathrm{Rat}\left(\left(1 \cdot \mathrm{Rat}\left(\left(\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k(\Gamma)}\right)^*\right)\right)\right)\right)(\mathrm{Rat}_k(\Gamma)). \tag{4.8}$$

Formellement, à chaque ensemble $R \in \mathrm{Rat}((\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k(\Gamma)})^*)$, nous associons une relation sur $\mathrm{Stacks}_k(\Gamma)$ notée $\mathcal{D}(R)$ et définie par:

$$\{(s,s') \in \mathrm{Stacks}_k(\Gamma) \times \mathrm{Stacks}_k(\Gamma) \mid \exists \theta \in \mathcal{R}(R), s \in \mathrm{Dom}(\theta) \text{ et } s' = \theta(s)\}.$$

En d'autres termes, l'application $\mathcal{D}$ est un morphisme de monoïdes entre le monoïde libre $(\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k(\Gamma)})^*$ et le monoïde (pour la composition) des relations sur $\mathrm{Stacks}_k(\Gamma)$.

Au niveau 1, ces relations sur les mots ont été étudiées par Caucal dans [Cau96, Cau03a, Cau03b] et sont connues sous le nom de relations préfixe-reconnaissables. Il montre en particulier que ces relations forment une algèbre de Boole et en donne une représentation normalisée.

Définition 4.5.1. Nous appellerons relation préfixe-reconnaissable de niveau k toute relation de $\mathcal{D}(\mathrm{Rat}((\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k(\Gamma)})^*))$ et nous noterons PR_k l'ensemble $\mathcal{D}(\mathrm{Rat}((\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k(\Gamma)})^*))$

Le sous-paragraphe 4.5.1 rappelle les propriétés des relations préfixe-reconnaissables de niveau 1 et fait le lien entre la présentation de Caucal et notre formalisme. Dans le sous-paragraphe 4.5.2, les propriétés des relations préfixe-reconnaissables de niveau 1 sont étendues à tout niveau. En particulier et quelque soit le niveau k, l'ensemble PR_k est une algèbre de Boole. Enfin dans le sous-paragraphe 4.5.3, nous présentons une dernière notion d'accepteurs finis pour les ensembles de $\mathrm{Rat}_{k+1}(\Gamma)$ basée sur les relations préfixe-reconnaissables de niveau k et sur l'équation 4.8.

4.5.1 Relations de PR_1.

Dans [Cau96], Caucal étudie la famille des relations sur les mots de Γ^* qui s'écrivent comme une union finie de relations de la forme $(U \times V) \cdot W$ où U,V et W sont des sous-ensembles rationnels de Γ^* et où le produit d'une relation $R \subseteq \Gamma^* \times \Gamma^*$ par un ensemble $P \subseteq \Gamma^*$ est une relation, notée $R \cdot P$, et égale

à $\{(u \cdot w,v \cdot w) \mid (u,v) \in R \text{ et } w \in P\}$. Ces relations correspondent à une généralisation des systèmes de récriture préfixe de mots. Comme nous avons pris la convention que le symbole de haut de pile correspond à la dernière lettre du mot[10], il nous faut considérer la version suffixe des relations préfixe-reconnaissables qui sont des unions finies de relations de la forme $W \cdot (U \times V)$ où U,V et W appartiennent à $\mathrm{Rat}(\Gamma^*)$. Cette variation n'a aucun impact sur les résultats obtenus dans [Cau96]. Nous allons établir l'équivalence entre les relations préfixe-reconnaissables sur Γ^* et les relations de $\mathrm{PR}_1(\Gamma)$.

Les relations préfixe-reconnaissables appartiennent à PR_1. En effet pour tout U,V et W appartenant à $\mathrm{Rat}(\Gamma^*)$, la relation $W \cdot (U \times V)$ est égale à $\mathcal{D}(\overline{U} \cdot T_W \cdot V)$ et appartient donc à PR_1. Comme par définition PR_1 est fermé par union, les relations préfixe-reconnaissables appartient bien à PR_1.

Pour l'inclusion réciproque, Caucal établit que les relations préfixe-reconnaissables sont fermées par union, concaténation et clôture transitive. Il suffit pour conclure de remarquer que pour tout $\gamma \in \Gamma$, $\mathcal{D}(\{\gamma\}) = \Gamma^* \cdot (\{\varepsilon\},\{\gamma\}) \in \mathrm{PR}_1$ et $\mathcal{D}(\{\bar{\gamma}\}) = \Gamma^* \cdot (\{\gamma\},\{\varepsilon\}) \in \mathrm{PR}_1$ et pour tout $W \in \mathrm{Rat}(\Gamma^*)$, $\mathcal{D}(\{T_W\}) = W \cdot (\{\varepsilon\},\{\varepsilon\})$.

Proposition 4.5.2 ([Cau96])**.** *Les relations préfixe-reconnaissables sur* Γ^* *sont les relations de* $PR_1(\Gamma)$.

Exemple 4.5.3. Pour tout $p \geq 2$ et pour tout $q \in [0,p-1]$, nous noterons $A_{p,q}$ (resp. $B_{p,q}$) l'ensemble rationnel contenant tous les mots $w \in a^*$ (resp. $w \in b^*$) tel que $|w| = p \mod q$. Considérons la relation R de PR_1 définie par:

$$\mathcal{D}\left(\left(\bar{b}^* \cdot \left(\bigcup_{i\in[0,2]} T_{A_{i,3}} b^* T_{a^* \cdot B_{i,3}}\right)\right)^+\right).$$

On vérifie facilement que R est égale à:

$$\bigcup_{i\in[0,2]} A_{i,3} \cdot (B^*,B_{i,3}).$$

Dans notre formalisme, l'égalité de la proposition 4.5.2 se traduit par la proposition suivante.

Proposition 4.5.4. *Toute relation de* $PR_1(\Gamma)$ *est égale à une union finie de relations dans l'ensemble* $\mathcal{D}(Rat(\overline{\Gamma}^*) \cdot \mathcal{T}_{Rat_1(\Gamma)} \cdot Rat(\Gamma^*))$.

Pour montrer la fermeture par complémentaire, Caucal montre que toute relation préfixe-reconnaissable peut s'écrire comme une union finie de relations de la

10. Et non la première comme dans [Cau96]

forme $W \cdot (U \times V)$ où U,V et $W \in \mathrm{Rat}(\Gamma^*)$ avec $\mathrm{First}(U) \cap \mathrm{First}(V) \subseteq \{\varepsilon\}$. Une telle relation est appelée clôture gauche d'une relation reconnaissable irréductible dans [Cau96].

Exemple 4.5.5. Reprenons la relation R de l'exemple 4.5.3. Il est aisé de vérifier que R s'écrit comme l'union finie suivante:

$$\bigcup_{i\in[0,2]} \left(\bigcup_{j+k=i} (A_{i,3} \cdot B_{j,3}) \cdot (\{\varepsilon\},B_{k,3}) \right) \cup (A_{i,3} \cdot B_{i,3}) \cdot (b^*,\{\varepsilon\}).$$

La propriété clé de ces relations est qu'elles correspondent à des ensembles de suites réduites d'instructions dans $\mathrm{Rat}((\Gamma_1 \cup \mathcal{T}_{\mathrm{Rat}_1(\Gamma)})^*)$. En particulier, pour tout couple $(u,v) \in W \cdot (U,V)$, il existe une unique décomposition $u = wu'$ et $v = wv'$ où $w \in W$, $u' \in U$ et $v' \in V$. L'unicité de cette décomposition permet d'obtenir la fermeture par complémentaire.

Proposition 4.5.6 ([Cau96])**.** *Les relations préfixe-reconnaissables forment une algèbre de Boole.*

4.5.2 Relations de PR_k.

Dans ce sous-paragraphe, nous étendons les propositions 4.5.4 et 4.5.6 aux relations de PR_k. Dans ce but, nous définissons, pour tout $k \geq 1$, une famille Rew_k d'ensembles de $\mathrm{Rat}((\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k})^*)$. Notre but sera, dans un premier temps, d'établir que toute relation de PR_k est égale à une union finie de relations dans $\mathcal{D}(\mathrm{Rew}_k)$ (cf. théorème 4.5.12).

Avant de définir la famille Rew_k, il nous faut introduire deux notations. Nous associerons, à chaque suite $\rho \in (\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k})^*$, un symbole dans $\Gamma_k^\circ \cup \{\varepsilon\}$ noté $\mathrm{First}(\rho)$ (resp. $\mathrm{Last}(\rho)$), et valant ε si $O = \{i \in [1,|\rho|] \mid \rho(i) \in \Gamma_k^\circ\}$ est égal à $\emptyset$ et valant $\rho(\min(O))$ (resp. $\rho(\max(O))$) sinon.

$$\begin{array}{lcl}
\mathrm{Norm}_1(\Gamma) & = & \mathrm{Rat}(\Gamma^*) \\
\mathrm{Rew}_1(\Gamma) & = & \{\, \overline{U} \cdot \mathcal{T}_W \cdot V \mid U,V \in \mathrm{Norm}_1(\Gamma), W \in \mathrm{Rat}_1(\Gamma), \\
 & & \qquad \mathrm{First}(U) \cap \mathrm{First}(V) \subseteq \{\varepsilon\} \,\} \\
\mathrm{Norm}_{k+1}(\Gamma) & = & \mathrm{Rew}_1(\Gamma) \cdot \mathrm{Rat}((1 \cdot \mathrm{Rew}_k(\Gamma))^*) \\
\mathrm{Rew}_{k+1}(\Gamma) & = & \{\, \overline{U} \cdot \mathcal{T}_W \cdot V \mid U,V \in \mathrm{Norm}_{k+1}(\Gamma), W \in \mathrm{Rat}_{k+1}(\Gamma), \\
 & & \qquad \mathrm{First}(U) \cap \mathrm{First}(V) \subseteq \{\varepsilon\}, \\
 & & \qquad \text{et } \overline{\mathrm{Last}(W)} \cap (\mathrm{First}(U) \cup \mathrm{First}(V)) \subseteq \{\varepsilon\} \,\}.
\end{array} \tag{4.9}$$

Pour pouvoir travailler avec les relations de PR_k, il nous faut fixer une représentation finie (et symbolique) de ces relations. Nous représentons de telles

relations par un automate sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$. Formellement, un automate sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ accepte la relation sur $\mathrm{Stacks}_k(\Gamma)$, notée $\mathcal{D}(A)$, définie par $\mathcal{D}(\mathcal{I}(A))$. Une adaptation immédiate de la proposition 4.3.5 établit que pour toute relation R de PR_k, il existe un automate A sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ tel que $\mathcal{D}(A) = R$.

4.5.2.1 Représentation normalisée des relations de PR_k.

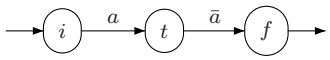
Dans ce sous paragraphe, nous établissons le théorème 4.5.12 qui établit que toute relation de PR_k est une union finie de relations dans $\mathcal{D}(\mathrm{Rew}_k)$.

La première étape est d'établir que les relations de PR_k sont acceptées par des automates réduits sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$. Pour cela, nous adaptons la preuve de la proposition 4.3.48.

Considérons un automate A sur Γ_k avec tests dans Rat_k. Comme dans le sous-paragraphe 4.3.3.2, nous allons éliminer les boucles des calculs de A. L'unique différence réside dans la définition des langages de boucles. Pour tout $p,q \in Q_A$, nous noterons $B_{p,q}$ l'ensemble des piles $s \in \mathrm{Stacks}_k(\Gamma)$ telles qu'il existe un calcul de A partant de la configuration (p,s) et arrivant à la configuration (q,s).

Remarque 4.5.7. Dans le sous-paragraphe 4.3.3.2, la notion de boucle utilisée est plus restrictive. Rappelons que nous disons l'automate A boucle sur la pile $s \in \mathrm{Stacks}_k(\Gamma)$ en partant dans l'état p et en revenant dans l'état q s'il existe un calcul $(p_0,s_0) \xrightarrow[A]{} \cdots \xrightarrow[A]{} (p_n,s_n)$ tel que $s_0 = s_n = s$, $p_0 = p$, $p_n = q$ et pour tout $\ell \in [0,n]$, $\rho_s \sqsubseteq \rho_{s_\ell}$. Pour tout $p,q \in Q_A$, nous notons $L_{p,q}$ l'ensemble des piles s de niveau k telles que A boucle sur s en partant de p et en revenant en q. Cette notion n'est pas suffisante lorsque l'on considère les relations acceptées par les automates sur Γ_k.

Considérons l'automate A sur Γ_1 présenté ci-contre. La relation acceptée est $\mathcal{D}(\bar{a}a) = \{(w,w) \mid w \in \Gamma^* a\}$. Le langage $L_{i,f}$ est vide alors que le langage $B_{i,f}$ est égal à $\Gamma^* a$ et nous obtenons donc une représentation réduite de $\mathcal{D}(A)$ qui est égale à $T_{\Gamma^* \cdot a}$.



Par une adaptation immédiate de la construction du lemme 4.3.46, nous montrons que pour tout $p,q \in Q_A$, le langage $B_{p,q}$ est accepté par un automate alternant sur Γ_k. Remarquons que l'automate alternant a le même niveau que l'automate A alors que dans le lemme 4.3.46 (où la notion de boucle est plus restreinte) l'automate alternant a un niveau de moins que celui de A.

Lemme 4.5.8. *Pour tout automate A sur Γ_k avec tests dans $Rat_k(\Gamma)$ et pour tout $p,q \in Q_A$, il existe un automate $A_{p,q}$ alternant sur Γ_k acceptant $B_{p,q}$.*

Démonstration. Soient $A = (Q_A,I_A,F_A,\mu_A,\Delta)$ un automate sur Γ_k avec tests dans

$\mathcal{L} = \{ L_1, \ldots, L_n \} \subset \mathrm{Rat}_k(\Gamma)$ et p_0, q_0 des états de Q_A.

Nous allons commencer par construire un automate $B = (Q_B, I_B, \Delta_B)$ alternant sur Γ_k acceptant B_{p_0,q_0}. L'ensemble des états Q_B est égal $Q_A \times Q_A \times (\Gamma_k^{\mathrm{o}} \cup \{ k, \bullet \}) \times \mathrm{Sing}(\Gamma_k^{\mathrm{T}}) \times 2^{\mathcal{L}}$ et l'ensemble des états initiaux I_B est $\{(p_0, q_0, \bullet, T, T') \mid T \in \mathrm{Sing}(\Gamma_k^{\mathrm{T}})$ et $T' \subseteq \mathcal{L}\}$. Pour définir l'ensemble des transitions Δ_B, nous considérons des transitions δ de la forme suivante:

$$\delta = (p,q,\gamma,T,L),T,L \rightarrow \bigwedge_{\gamma' \in R} \quad \bigwedge_{(p',q',\gamma',T',L') \in Q_{\gamma'}} ((p',q',\gamma',T',L'),\gamma')$$

où $R \subseteq (\Gamma_k^{\mathrm{o}} \cup \{ k \}) \setminus \{ \bar{\gamma} \}$ si $\gamma \neq \bullet$ et $R \subseteq (\Gamma_k^{\mathrm{o}} \cup \{ k \})$ si $\gamma = \bullet$ et pour tout $\gamma' \in R$, $Q_{\gamma'} \subseteq Q_B$. Pour tout $\gamma' \in R$, nous définissons un ensemble $R_{\gamma'} \subseteq Q_A \times Q_A$ égal à:

$$\begin{aligned} R_{\gamma'} \quad = \quad & \{(p,q) \mid \exists (r,t,\gamma,T',L') \in Q_\gamma, p \xrightarrow{\gamma'} r, T_1, L_1 \in \Delta_A, t \xrightarrow{\bar{\gamma}'} q, T_2, L_2 \in \Delta_A, \\ & T_1 \leq T, T_2 \leq T', L_1 \subseteq L \text{ et } L_2 \subseteq L'\}. \end{aligned}$$

La transition δ appartient à Δ_B si $(p,q) \in \left(\bigcup_{\gamma' \in R} R_{\gamma'} \right)^*$. Par la proposition 4.3.38, il existe un automate C alternant sur Γ_k équivalent à B. □

Grâce au lemme précédent, nous pouvons maintenant réduire les automates sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ sans changer la relation acceptée. La construction est une adaptation de la construction de la proposition 4.3.48.

Proposition 4.5.9. *Pour tout $k \geq 1$, pour tout automate A sur Γ_k avec tests dans $\mathcal{L} \subseteq Rat_k(\Gamma)$, il existe un automate B réduit sur Γ_k avec tests dans $\mathcal{L}' \subseteq Rat_k(\Gamma)$.*

Démonstration. Soit $A = (Q_A, I_A, F_A, \mu_A, \Delta_A)$ un automate sur Γ_k avec tests dans $\mathcal{L} \subseteq \mathrm{Rat}_k$. Nous construisons un automate $B = (Q_B, I_B, F_B, \mu_B, \Delta_B)$ réduit sur Γ_{k+1} avec tests dans un ensemble fini $\mathcal{L} \subset \mathrm{Rat}_{k+1}$ équivalent à A.

L'ensemble $\mathcal{L}$ contient les langages acceptés par les automates $A^\gamma_{p,q}$ pour tout $p,q \in Q_A$ et $\gamma \in \Gamma_k^{\mathrm{o}} \cup \{ k \}$. A ces langages, il faut ajouter les langages de tests associés aux états initiaux. Pour tout état $p \in Q_A$, nous noterons S_p le langage $\bigcup_{i \in I_A} \mathcal{S}(A_{i,p}) \cap (\bigcap_{L \in \mu_A(i)} L)$. Par la proposition 4.3.16 et le théorème 4.3.43, le langage S_p appartient à Rat_{k+1}. Nous prenons donc $\mathcal{L}$ égal à:

$$\begin{aligned} \mathcal{L} \quad = \quad & \{\mathcal{S}(A^\gamma_{p,q}) \mid p,q \in Q_A \text{ et } \gamma \in \Gamma_k^{\mathrm{o}} \cup \{ k \}\} \\ & \cup \{S_q \mid q \in Q_A\} \end{aligned}$$

où pour tout $p,q \in Q_A$ et $\gamma \in \Gamma_{k+1}^{\mathrm{o}}$, $A^\gamma_{p,q}$ est l'automate alternant sur Γ_k défini dans le lemme 4.3.46. Tout langage $L \in \mathcal{L}$ est accepté par un automate alternant sur Γ_k de taille polynomiale en $|A|$.

Nous pouvons maintenant définir l'automate B. L'ensemble des états Q_B est égal à $Q_A \times (\Gamma_{k+1}^{\circ} \cup \{\varepsilon\})$. Les ensembles d'états initiaux et finaux sont respectivement $I_B = Q_A \times \{\varepsilon\}$ et $F_B = F_A \times (\Gamma_{k+1}^{\circ} \cup \{\varepsilon\})$. L'application μ_B est définie pour tout $(q,\varepsilon) \in Q_B$ par $\mu_B((q,\varepsilon)) = S_q$. L'ensemble des transitions Δ_B est donné par:

$$\Delta_B = \{(p,\gamma') \xrightarrow{\gamma} ((r,\gamma),\gamma),T,L \cup T_{\mathcal{S}(A^{\gamma}_{q,r})} \mid \gamma \in \Gamma_k^{\circ} \cup \{k\}, \gamma \neq \bar{\gamma}' \text{ et } p \xrightarrow{\gamma} q,T,L\}.$$

Par construction l'automate B est réduit. Une adaptation immédiate de la preuve de la proposition 4.3.48 établit que $\mathcal{D}(B) = \mathcal{D}(A)$. □

L'étape suivante consiste à exprimer la relation acceptée par un automate réduit sur Γ_k avec tests dans Rat_k comme une union finie de relations représentées par des ensembles d'instructions dont la «forme» correspond aux ensembles de Rew_k.

Proposition 4.5.10. *Pour tout automate A réduit sur Γ_{k+1} avec des langages de tests dans Rat_{k+1}, la relation $\mathcal{D}(\mathcal{R})$ s'écrit comme une union finie de relations de la forme $\mathcal{D}(\overline{U} \cdot \mathcal{T}_W \cdot V)$ où U et V sont des ensembles réduits dans $Rat((\Gamma_k \cup \{k\} \cup \mathcal{T}_{Rat_{k+1}(\Gamma)})^*)$ et où W appartient à $Rat_{k+1}(\Gamma)$ avec:*
$First(U) \cap First(V) \subseteq \{\varepsilon\}$ et $\overline{Last(W)} \cap (First(U) \cup First(V)) \subseteq \{\varepsilon\}$.

Démonstration. Soit $A = (Q_A,I_A,F_A,\mu_A,\Delta_A)$ un automate réduit sur Γ_{k+1} avec tests dans $\mathrm{Rat}_{k+1}(\Gamma)$. Pour tout $p \in Q_A$ et pour tout $\gamma \in \Gamma_k^{\circ} \cup \{\bar{k},\varepsilon\}$, nous définissons U_p^{γ} comme le langage des suites d'instructions acceptées par l'automate $(Q_A,I_A,\{p\},\mu_A,\Delta_A)$ restreint aux suites d'instructions ρ ne contenant pas l'instruction k et telle que $\mathrm{Last}(\rho) = \gamma$. De manière analogue, nous définissons, pour tout $p \in Q_A$ et pour tout $\gamma \in \Gamma_k^{\circ} \cup \{k,\varepsilon\}$, V_p^{γ} comme le langage des suites d'instructions acceptées par l'automate $(Q_A,\{p\},F_A,\Delta_A)$ restreint aux suites d'instructions ρ ne contenant pas l'instruction $\bar{k}$ et telle que $\mathrm{First}(\rho) = \gamma$.

Comme A est un automate réduit, il est aisé de voir que pour tout $q \in Q_A$ et pour tout $\gamma \in \Gamma_k$, $\overline{U}_p^{\gamma}$ et V_p^{γ} sont des ensembles réduits dans $\mathrm{Rat}((\Gamma_k \cup \{k\} \cup \mathcal{T}_{\mathrm{Rat}_{k+1}(\Gamma)})^*)$. Nous affirmons que:

$$\mathcal{D}(A) = \bigcup_{p \in Q_A} \bigcup_{(\gamma_1,\gamma_2,\gamma_3) \in I} \mathcal{D}\left(U_p^{\gamma_1} \cdot \mathcal{T}_{W^{\gamma_2}} \cdot V_p^{\gamma_3}\right)$$

où I est le sous-ensemble de $\Gamma_k^{\circ} \cup \{\bar{k},\varepsilon\} \times \Gamma_k^{\circ} \cup \{k,\varepsilon\} \times \Gamma_k^{\circ} \cup \{\bar{k},\varepsilon\}$ défini par:

$$(\gamma_1,\gamma_2,\gamma_3) \in I \Leftrightarrow \{\gamma_1\} \cap \{\bar{\gamma}_3\} \subseteq \{\varepsilon\} \text{ et } \{\gamma_2\} \cap \{\gamma_1,\bar{\gamma}_3\} \subseteq \{\varepsilon\}.$$

L'inclusion réciproque est immédiate. Pour l'inclusion directe, considérons deux piles s et s' dans $\mathrm{Stacks}_{k+1}(\Gamma)$ telles que $(s,s') \in \mathcal{D}(A)$. Notons ρ_s et $\rho_{s'}$

les suites réduites des piles s et s'. Notons r la pile de $\mathrm{Stacks}_{k+1}(\Gamma)$ dont la suite réduite est le plus petit préfixe commun de ρ_s et $\rho_{s'}$.

Comme (s,s') appartient à $\mathcal{D}(A)$, il existe $\rho \in \mathcal{I}(A)$ tel que $s' = \mathcal{R}(\rho)(s)$. Il existe deux suites d'instructions ρ_1 et $\rho_2 \in \Gamma_{k+1}^*$ telles que $\rho = \rho_1 \cdot \rho_2$ et $\mathcal{R}(\rho_1)(s) = t$ et $\mathcal{R}(\rho_2)(t) = s'$. Comme $\rho \in \mathcal{I}(A)$, il existe un état $q \in Q_A$ tel que $i \xrightarrow[A]{\rho_1} q$ et $q \xrightarrow[A]{\rho_2} f$ où $i \in I_A$ et $f \in F_A$. Notons $\gamma_1 = \mathrm{Last}(\rho_1)$, $\gamma_2 = \mathrm{Last}(\rho_r)$ et $\gamma_3 = \mathrm{First}(\rho_2)$. La suite d'instructions ρ_1 appartient à $U_q^{\gamma_1}$ et la suite ρ_2 appartient à $V_q^{\gamma_3}$ et $r \in W^{\gamma_2}$. Comme $\rho = \rho_1 \cdot \rho_2$ est une suite réduite, $\{\,\overline{\gamma_1}\,\} \cap \{\,\gamma_3\,\} \subseteq \{\,\varepsilon\,\}$. Comme $\rho_r\overline{\rho_1}$ est la suite réduite de s, $\{\,\gamma_2\,\} \cap \{\,\gamma_1\,\} \subseteq \{\,\varepsilon\,\}$. Comme $\rho_r\rho_2$ est la suite réduite de s', $\{\,\gamma_2\,\} \cap \{\,\bar{\gamma}_3\,\} \subseteq \{\,\varepsilon\,\}$. Il suit donc $(\gamma_1,\gamma_2,\gamma_3) \in I$. Comme $(s,s') \in \mathcal{D}\left(U_p^{\gamma_1} \cdot \mathcal{T}_{W^{\gamma_2}} \cdot V_p^{\gamma_3}\right)$, nous avons établi l'inclusion directe. □

La proposition suivante est l'étape clé de la preuve.

Proposition 4.5.11. *Pour tout automate A réduit sur Γ_{k+1} avec tests dans $\mathcal{L} \subset Rat_{k+1}(\Gamma)$ ne contenant pas l'instruction $\bar{k}$, la relation $\mathcal{D}(A)$ est égale à une union finie de relations de la forme $\mathcal{D}(\mathcal{T}_{Rat_{k+1}} \cdot \mathcal{I}(B))$ où B est un automate réduit sur Γ_{k+1} avec tests dans $Rat_k(\Gamma)$ et n'utilisant pas l'instruction $\bar{k}$.*

Démonstration. Soit $A = (Q_A,I_A,F_A,\mu_A,\Delta_A)$ un automate réduit sur Γ_{k+1} avec tests dans $\mathcal{L} = \{\,L_1,\ldots,L_n\,\} \subset \mathrm{Rat}_{k+1}(\Gamma)$. Pour tout $\ell \in [1,n]$, il existe un automate $A_\ell = (Q_\ell,\{\,i_\ell\,\},F_\ell,\Delta_\ell)$ déterministe et complet avec tests dans $\mathcal{L}_i \subset \mathrm{Rat}_k(\Gamma)$ acceptant L_ℓ (cf. théorème 4.4.15). Pour tout $\ell \in [1,n]$ et pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$, nous noterons $A_\ell(s)$ l'unique état q tel que qu'il existe un calcul de A_ℓ partant de la configuration $(i_\ell, [\,]_{k+1})$ et arrivant dans la configuration (q,s). Pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$, nous noterons X_s le uplet $(q_1,\ldots,q_n) \in \prod_{\ell\in[1,n]} Q_\ell$ défini pour tout $\ell \in [1,n]$, $q_\ell = A_\ell(s)$.

Pour tout uplet $d \in \prod_{\ell\in[1,n]} Q_\ell$, l'ensemble S_d des piles s de niveau $k+1$ telle que $X_s = d$ appartient à $\mathrm{Rat}_{k+1}(\Gamma)$. En effet, S_d est l'intersection pour tout $\ell \in [1,n]$ des $\{s \mid A_\ell(s) = q_\ell\} \in \mathrm{Rat}_{k+1}(\Gamma)$.

Pour tout uplet $d,d' \in \prod_{\ell\in[1,n]} Q_\ell$ et pour tout $\gamma \in \Gamma_k^o \cup \{\,k\,\}$, il existe un ensemble $R_{d,d'}^\gamma$ dans $\mathrm{Rat}_k(\Gamma)$ tel que pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ telle que $X_s = d$ et telle que $s' = \mathcal{R}(\gamma)(s)$ est défini, $X_{s'} = d'$ si et seulement si $\mathrm{top}_k(s') \in R_{d,d'}^\gamma$.
Si $d = (q_1,\ldots,q_n)$ et $d' = (q'_1,\ldots,q'_n)$, il suffit de prendre:

$$R_{d,d'}^\gamma = \bigcap_{\ell\in[1,n]} \quad \bigcup_{q_\ell \xrightarrow{\gamma} q'_\ell, T_R \in \Delta_\ell} R.$$

Pour tout $d \in \prod_{\ell\in[1,n]} Q_\ell$, nous allons définir un automate B_d réduit sur Γ_{k+1} avec tests dans $\mathrm{Rat}_k(\Gamma)$ qui satisfasse pour tout $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ telle que $X_s = d$, $\mathcal{D}(A)(s) = \mathcal{D}(B_d)(s)$.

Soit $d_0 = (q_1,\ldots,q_n) \in \prod_{\ell\in[1,n]} Q_\ell$. Nous construisons l'automate $B_{d_0} = (Q_B,I_B,F_B,\Delta_B)$. Nous prenons $Q_B = Q_A \times (\prod_{\ell\in[1,n]} Q_\ell)$, I_B est égal à l'ensemble:

$$\{(i,d_0) \mid i \in I_A \text{ et } \forall \ell \in [1,n], L_\ell \in \mu_A(i) \Rightarrow q_\ell \in F_\ell\}$$

et $F_B = F_A \times (\prod_{\ell\in[1,n]} Q_\ell)$. L'ensemble des transitions Δ_ℓ est défini par:

$$\{(p,d) \xrightarrow{\gamma} (q,\gamma,d'), R^\gamma_{d,d'} \mid d,d' = (q'_1,\ldots,q'_n) \in (\prod_{\ell\in[1,n]} Q_\ell) \text{ et } p \xrightarrow{\gamma} q, T \in \Delta_A \text{ et } \forall \ell \in [1,n], L_\ell \in T \to q'_\ell \in F_\ell\}.$$

Comme A est réduit, B l'est aussi. De plus par construction, nous avons que pour toutes piles $s,s' \in \mathrm{Stacks}_{k+1}(\Gamma)$ telles que $X_s = d_0$ alors $((i,d_0),s) \xrightarrow[B_{d_0}]{\rho} ((q,d),s')$ si et seulement si $(i,s) \xrightarrow[A]{\rho} (q,s')$ et $X_{s'} = d$. Il s'en suit donc que B_{d_0} satisfait la propriété annoncée. Nous avons donc:

$$\mathcal{D}(A) = \bigcup_{d\in\prod_{\ell\in[1,n]} Q_\ell} \mathcal{D}(S_d \cdot \mathcal{I}(B_d)).$$

□

Nous pouvons maintenant établir le résultat de normalisation pour les relations de PR_k.

Théorème 4.5.12. *Pour tout niveau $k \geq 1$ et pour tout alphabet fini Γ, toute relation $R \in PR_k(\Gamma)$ est une union finie de relations $\mathcal{D}(Rew_k(\Gamma))$.*

Démonstration. La preuve procède par récurrence sur le niveau $k \geq 1$.
Cas de base : $k = 1$. Au niveau 1, cette propriété a été établie par Caucal dans [Cau96] (cf. proposition 4.5.4). Pour être complet, nous en redonnons une preuve. Soit A un automate sur Γ_1 avec tests dans Rat_1. Par les propositions 4.5.9, 4.5.10 et 4.5.11, la relation R est égale à une union finie de relations de la forme $\overline{U}\cdot\mathcal{T}_W\cdot V$ où U,V appartiennent à $\mathrm{Rat}((\Gamma_k \cup \{k\} \cup \mathcal{T}_{\mathrm{Rat}_{k+1}(\Gamma)})^*))$ avec $\mathrm{First}(U)\cap First(V) \subseteq \{\varepsilon\}$. Remarquons que la relation $\overline{\mathrm{Last}(W)} \cap (\mathrm{First}(U) \cup \mathrm{First}(V)) \subseteq \{\varepsilon\}$ est trivialement vérifiée car $\mathrm{Last}(W)$, $\mathrm{First}(U)$ et $\mathrm{First}(V)$ sont des sous-ensembles de $\Gamma \cup \{\varepsilon\}$.
Etape de récurrence. Soient R une relation de PR_{k+1} et A un automate sur Γ_{k+1} avec tests dans $\mathrm{Rat}_{k+1}(\Gamma)$ acceptant R. Par les propositions 4.5.9, 4.5.10 et 4.5.11, la relation R est égale à une union finie de relations de la forme $\overline{U}\cdot\mathcal{T}_W\cdot V$ où U,V appartiennent à $\mathrm{Rat}((\Gamma_k \cup \{k\} \cup \mathcal{T}_{\mathrm{Rat}_{k+1}(\Gamma)})^*))$ avec $\mathrm{First}(U) \cap First(V) \subseteq \{\varepsilon\}$ et $\overline{\mathrm{Last}(W)} \cap (\mathrm{First}(U) \cup \mathrm{First}(V)) \subseteq \{\varepsilon\}$. Par hypothèse de récurrence, toute relation $\mathrm{Rat}((\Gamma_k \cup \{k\} \cup \mathcal{T}_{\mathrm{Rat}_{k+1}(\Gamma)})^*))$ est égale à une union finie de relations dans $\mathrm{Rew}_k(\Gamma)\cdot\mathrm{Rat}((k\cdot\mathrm{Rew}_k(\Gamma))^*)$. □

Une première conséquence de ce résultat est que le domaine et l'image des relations $\mathrm{PR}_k(\Gamma)$ sont des ensembles de $\mathrm{Rat}_k(\Gamma)$.

Proposition 4.5.13. *Pour toute relation $R \in PR_k(\Gamma)$, le domaine et l'image de R sont des ensembles de Rat_k.*

Démonstration. Soit R une relation dans $\mathrm{PR}_k(\Gamma)$. Par définition de $\mathrm{PR}_k(\Gamma)$, il existe un ensemble $M \in \mathrm{Rat}((\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k})^*)$ tel que $R = \mathcal{D}(M)$. Il est aisé de voir que l'image de R est égale à $\mathcal{R}(\Gamma_k^* \cdot M)([\ ]_k))$. D'après le corollaire 4.3.44, l'image de R appartient bien à Rat_k. Pour le domaine de R, il suffit de remarquer que $\mathrm{Dom}(\mathcal{D}(R)) = \mathrm{Im}(\mathcal{D}(\overline{R}))$. □

4.5.2.2 Propriétés des relations de $\mathbf{PR}_k$

Dans ce sous-paragraphe, nous établissons que pour tout $k \geq 1$, et pour tout alphabet fini, l'ensemble PR_k est une algèbre de Boole. Par définition, $\mathrm{PR}_k(\Gamma)$ est fermé par union et contient la relation vide. Il faut donc montrer que $\mathrm{PR}_k(\Gamma)$ est fermé par complémentaire (*i.e.* pour tout $R \in \mathrm{PR}_k$, $R^{\mathrm{c}} = (\mathrm{Stacks}_k(\Gamma) \times \mathrm{Stacks}_k(\Gamma)) \setminus R$ appartient à PR_k). La preuve de la fermeture par complémentaire est un adaptation de la preuve de Caucal [Cau96] pour le niveau 1. Elle repose sur le théorème 4.5.12 et sur la propriété suivante qui généralise l'unicité de la décomposition mentionnée dans le sous-paragraphe 4.5.1.

Lemme 4.5.14. *Pour toute relation $R = \overline{U} \cdot T_W \cdot V$ de Rew_k avec U,$V \in Norm_k$ et $W \in Rat_k(\Gamma)$ et pour toutes piles r et $s \in \mathrm{Stacks}_k(\Gamma)$ telles que (s,s') appartienne à $\mathcal{D}(R)$, il existe une unique pile $w \in \mathrm{Stacks}_k(\Gamma)$ telle que $w \in W$, $(w,s) \in \mathcal{D}(U)$ et $(w,t) \in \mathcal{D}(V)$.*
De plus, la pile w est la pile de niveau k dont la suite réduite d'instructions est le plus petit préfixe commun des suites réduites de r et s.

Démonstration. Soit une relation $R = \overline{U} \cdot T_W \cdot V$ de Rew_k avec U,$V \in \mathrm{Norm}_k$ et $W \in \mathrm{Rat}_k(\Gamma)$. Par définition de Rew_k, nous avons $\mathrm{First}(U) \cap \mathrm{First}(V) \subseteq \{\, \varepsilon \,\}$ et $\overline{\mathrm{Last}(W)} \cap (\mathrm{First}(U) \cup \mathrm{First}(V)) \subseteq \{\, \varepsilon \,\}$. Soient r et s deux piles de $\mathrm{Stacks}_k(\Gamma)$ telles que (r,s) appartienne à $\mathcal{D}(R)$.

L'existence de la pile w est immédiate. Montrons que cette pile est unique. Supposons qu'il existe deux piles w_1 et $w_2 \in W$ telles que (w_1,r) et (w_2,r) appartiennent à $\mathcal{D}(U)$ et telles que (w_1,s) et (w_2,s) appartiennent à $\mathcal{D}(V)$. Nous noterons ρ_r, ρ_s, ρ_{w_1} et ρ_{w_2} les suites réduites respectives des piles r, s, w_1 et w_2.

Comme U est un ensemble de suites d'instructions réduites et que $\overline{\mathrm{Last}(W)} \cap \mathrm{First}(U) \subseteq \{\, \varepsilon \,\}$, nous avons $\rho_s = \rho_{w_1} \cdot \rho_s^1 = \rho_{w_2} \cdot \rho_s^2$ pour deux suites d'instructions ρ_s^1 et ρ_s^2. Par un argument similaire, nous avons $\rho_r = \rho_{w_1} \cdot \rho_r^1 = \rho_{w_2} \cdot \rho_r^2$ pour deux suites d'instructions ρ_r^1 et ρ_r^2.

De plus, nous savons que les suites ρ_s^1 et ρ_s^2 appartiennent à $\tilde{U}$ (où $\tilde{U}$ est obtenu en effaçant dans toutes les suites U les occurrences des instructions de tests). De même, nous avons que les suites ρ_s^1 et ρ_s^2 appartiennent à $\tilde{V}$. Comme $\mathrm{First}(U) = \mathrm{First}(\tilde{U})$ et $\mathrm{First}(V) = \mathrm{First}(\tilde{V})$ et que $\mathrm{First}(U) \cap \mathrm{First}(V) \subseteq \{\varepsilon\}$, il s'en suit que ρ_{w_1} et ρ_{w_2} sont égales au plus petit préfixe commun de ρ_s et ρ_r. Il en résulte que w_1 et w_2 sont égales. □

Avant d'établir la fermeture par complémentaire de PR_k, nous montrons que les relations de $\mathcal{D}(\mathrm{Rew}_k)$ sont closes par intersection.

Lemme 4.5.15. *Pour tout niveau $k \geq 1$ et pour toutes R_1 et R_2 dans $\mathcal{D}(Rew_k)$, la relation $R_1 \cap R_2$ appartient à $\mathcal{D}(Rew_k)$.*

Démonstration. La preuve procède par récurrence sur le niveau k.
Cas de base : $k = 1$**.** Soient $R_1 = \overline{U}_1 \cdot T_{W_1} \cdot V_1$ et $R_2 = \overline{U}_2 \cdot T_{W_2} \cdot V_2$ appartenant à Rew_1. Nous affirmons que la relation $R = (U_1 \cap U_2) \cdot T_{W_1 \cap W_2} \cdot (V_1 \cap V_2)$ de Rew_1 est telle que $\mathcal{D}(R) = \mathcal{D}(R_1) \cap \mathcal{D}(\mathcal{R}_2)$. L'inclusion directe est immédiate. Pour l'inclusion réciproque, soit (r,s) un couple de piles de $\mathrm{Stacks}_k(\Gamma)$ appartenant à $\mathcal{D}(R_1)$ et $\mathcal{D}(R_2)$. Soit w la pile de $\mathrm{Stacks}_k(\Gamma)$ dont la suite réduite est le plus petit préfixe commun des suites réduites des piles s et r. Par le lemme 4.5.14, nous avons que $w \in W_1 \cap W_2$, $(w,r) \in \mathcal{D}(U_1) \cap \mathcal{D}(U_2)$ et $(w,s) \in \mathcal{D}(V_1) \cap \mathcal{D}(V_1)$. Il suit donc que (r,s) appartient à $\mathcal{D}(R)$.
Etape de récurrence. Par hypothèse de récurrence, nous savons que pour tous R_1 et R_2 appartenant à Rew_k, il existe $R \in \mathrm{Rew}_k$ tel que $\mathcal{D}(R)$. Pour être concis, nous désignerons $R_1 \Cap R_2$ un élément de Rew_k satisfaisant cette propriété.

Nous allons tout d'abord montrer que pour tous R_1 et R_2 dans Norm_{k+1}, la relation $\mathcal{D}(R_1) \cap \mathcal{D}(R_2)$ appartient aussi à $\mathcal{D}(\mathrm{Norm}_{k+1})$. Pour $i \in \{1,2\}$, comme R_i appartient à Norm_{k+1}, R_i est égal à $P_i \cdot M_i$ avec $P_i \in \mathrm{Rew}_k$ et $M_i \in \mathrm{Rat}((1 \cdot \mathrm{Rew}_k)^*)$. Enfin, soit $A_i = (Q_i,I_i,F_i,\Delta_i)$ un automate étiqueté par $1 \cdot \mathrm{Rew}_k(\Gamma)$ acceptant M_i.

Considérons l'automate $C = (Q_1 \times Q_2, I_1 \times I_2, F_1 \times F_2, \Delta_C)$ dont l'ensemble des transitions est donné par:

$$\{(p_1,p_2) \xrightarrow{1\cdot(N_1 \Cap N_2)} (q_1,q_2) \mid p_1 \xrightarrow{1\cdot N_1} q_1 \in \Delta_1 \text{ et } p_2 \xrightarrow{1\cdot N_2} q_2 \in \Delta_2\}.$$

Montrons maintenant que $\mathcal{D}(\mathrm{Rew}_k)$ est fermé par intersection. Soient $R_1 = \overline{U}_1 \cdot T_{W_1} \cdot V_1$ et $R_2 = \overline{U}_2 \cdot T_{W_2} \cdot V_2$ appartenant à Rew_{k+1} où U_1,U_2,V_1 et V_2 appartiennent à Norm_{k+1} et où W_1 et W_2 appartiennent à $\mathrm{Rat}_{k+1}(\Gamma)$. Nous affirmons que la relation $R = (U_1 \Cap U_2) \cdot T_{W_1 \cap W_2} \cdot (V_1 \cap V_2)$ de Rew_1 est telle que $\mathcal{D}(R) = \mathcal{D}(R_1) \cap \mathcal{D}(\mathcal{R}_2)$. La preuve est similaire à celle du niveau 1. □

Nous pouvons maintenant établir la fermeture par complémentaire de PR_k.

Théorème 4.5.16. *Pour tout $k \geq 1$ et pour tout alphabet fini Γ, l'ensemble PR_k est fermé par union, complémentaire, clôture transitive, inverse et restriction en image ou en domaine par un ensemble de $Rat_k(\Gamma)$.*

Démonstration. Excepté la fermeture par complémentaire, toutes ces propriétés découlent de la définition de PR_k. Pour la fermeture par complémentaire de PR_k, il suffit de monter que le complémentaire d'une relation dans $\mathcal{D}(\mathrm{Rew}_k)$ est égal à une union finie de relations dans $\mathcal{D}(\mathrm{Rew}_k)$.
En effet, considérons une relation $R \in \mathrm{PR}_k$. Par le théorème 4.5.12, R est égal à une union finie $\bigcup_{i\in I} R_i$ de relations de $\mathcal{D}(\mathrm{Rew}_k)$. Le complémentaire de R est égal à $\bigcap_{i\in I} R_i^{\mathrm{c}}$. Si nous avons établi pour tout $i \in I$ que R_i^{c} est égal à une union finie $\bigcup_{j\in I_j} R_i^j$ de relations dans Rew_c, il suit alors par les lois de Morgan que R^{c} est égal à une union finie d'intersections finies de relations de $\mathcal{D}(\mathrm{Rew}_k)$. Le lemme 4.5.15 permet alors de conclure que R^{c} appartient à PR_k.

Nous établissons maintenant par récurrence sur le niveau k que le complémentaire d'une relation dans $\mathcal{D}(\mathrm{Rew}_k)$ s'exprime comme une union finie de relation dans $\mathcal{D}(\mathrm{Rew}_k)$.
Cas de base : $k = 1$**.** La propriété a été établie par Caucal [Cau96] (cf. 4.5.6).
Etape de récurrence. Avant de passer à l'étape de récurrence, il nous faut fixer quelques notations. Nous noterons N l'ensemble de Norm_{k+1} contenant toutes les suites réduites sur $(\Gamma_k^{\circ} \cup \{\, k \,\})^*$. Pour tout $\gamma \in \Gamma_k^{\circ} \cup \{\, k \,\}$ et pour tout $W \in \mathrm{Rat}_{k+1}$, nous noterons W_γ l'ensemble W restreint aux piles s de $\mathrm{Stacks}_{k+1}(\Gamma)$ et que $\mathrm{Last}(s) = \gamma$. Pour tout $\gamma \in \Gamma_k^{\circ} \cup \{\, k \,\}$ et pour tout $U \in \mathrm{Norm}_{k+1}$, nous noterons U_γ l'ensemble W restreint à l'ensemble des suites d'instructions telles que $\mathrm{First}(\gamma)$ de $\mathrm{Stacks}_{k+1}(\Gamma)$ telles que $\mathrm{Last}(s) = \gamma$. Enfin pour toute relation $R \in \mathrm{Rew}_k$, nous noterons $R^{\Subset}$ un sous-ensemble de Rew_k tel que $\mathcal{D}(R)^{\mathrm{c}} = \bigcup_{R'\in\mathcal{R}^{\Subset}} \mathcal{D}(R')$. L'existence de $\mathcal{R}^{\Subset}$ est garantie par l'hypothèse de récurrence.

Notre but est d'établir que le complémentaire d'une relation dans $\mathcal{D}(\mathrm{Rew}_{k+1})$ appartient à $\mathrm{PR}(\Gamma_{k+1})$. Pour cela, nous allons tout d'abord montrer que pour toute relation dans $R \in \mathcal{D}(\mathrm{Norm}_{k+1})$, la relation $N \setminus R$ est égale à une union finie de relations dans $\mathcal{D}(\mathrm{Norm}_{k+1})$. Nous noterons $R^{\Subset}$ cet ensemble fini de relations.

Soit $R = P \cdot M$ un élément de Norm_{k+1} avec $P \in \mathrm{Rew}_k$ et $M \in \mathrm{Rat}((1 \cdot \mathrm{Rew}_k)^*)$. Soit $A = (Q_A, I_A, F_A, \Delta_A)$ un automate étiqueté par $1 \cdot \mathrm{Rew}_k$ acceptant M. Comme par hypothèse de récurrence PR_k forme un algèbre de Boole, nous pouvons supposer que A est tel que:

- pour tout $p,q,q' \in Q_A$, $p \xrightarrow{1\cdot R_1} q \in \Delta_A$, $p \xrightarrow{1\cdot R_2} q' \in \Delta_A$ et $q \neq q'$ implique $\mathcal{D}(R_1) \cap \mathcal{D}(R_2) = \emptyset$
- pour tout $p \in Q_A$, $\bigcup_{p \xrightarrow{1\cdot R} q \in \Delta_A} \mathcal{D}(R) = \mathrm{Stacks}_k(\Gamma) \times \mathrm{Stacks}_k(\Gamma)$.

Pour une construction détaillée de l'automate A, nous référons le lecteur à la proposition 4.4.24. Considérons l'automate B obtenu à partir de A en remplaçant

l'ensemble des états finaux F_A par son complémentaire $Q_A \setminus F_A$. Nous vérifions aisément que:

$$\mathcal{D}(N) \setminus \mathcal{D}(R) = \bigcup_{V \in P^{\mathrm{c}}} V \cdot (1 \cdot \mathrm{Rew}_k)^* \cup V \cdot \mathcal{L}(B).$$

Soit $R = \overline{U} \cdot T_W \cdot V$ une relation de Rew_{k+1} avec $U,V \in \mathrm{Norm}_{k+1}$ et $W \in \mathrm{Rat}_{k+1}(\Gamma)$. Nous allons montrer que le complémentaire de R est égal à une union finie de relations dans $\mathcal{D}(\mathrm{Rew}_{k+1})$.

Considérons l'ensemble C qui est une union finie de relations dans Rew_{k+1} défini par:

$$\begin{array}{rcl} C & = & \bigcup_{(\gamma_1,\gamma_2,\gamma_3)\in I} \quad \overline{N}_{\gamma_1} \cdot T_{W^{\mathrm{c}}_{\gamma_2}} \cdot N_{\gamma_3} \\ & \cup & \bigcup_{(\gamma_1,\gamma_2,\gamma_3)\in I} \bigcup_{U' \in U^{\Subset}} \quad \overline{U'}_{\gamma_1} \cdot T_{W_{\gamma_2}} \cdot N_{\gamma_3} \\ & \cup & \bigcup_{(\gamma_1,\gamma_2,\gamma_3)\in I} \bigcup_{V' \in U^{\Subset}} \quad \overline{N}_{\gamma_1} \cdot T_{W_{\gamma_2}} \cdot V_{\gamma_3} \end{array}$$

où I est l'ensemble des triplets $(\gamma_1,\gamma_2,\gamma_3) \in (\Gamma_k^{\mathrm{o}} \cup \{\, k,\varepsilon \,\})^3$ tels que $\{\, \gamma_1 \,\} \cap \{\, \gamma_3 \,\} \subseteq \{\, \varepsilon \,\}$ et tels que $\{\, \overline{\gamma_2} \,\} \cap \{\, \gamma_1,\gamma_3 \,\} \subseteq \{\, \varepsilon \,\}$.

On vérifie aisément que les éléments $P \in \mathrm{Rew}_{k+1}$ intervenant dans C sont tels que $\mathcal{D}(P)$ est inclus dans $\mathcal{D}(R)^{\mathrm{c}}$. Il suit donc que $\mathcal{D}(P) \subseteq \mathcal{D}(R)^{\mathrm{c}}$. Pour l'inclusion réciproque, considérons deux piles r et $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ telles que $(r,s) \notin \mathcal{D}(R)$. Notons w la pile de niveau $k+1$ dont la suite réduite est le plus petit préfixe commun des suites réduites de r et de s. Il existe donc deux suites d'instructions ρ'_s et ρ'_r telles que $\rho_r = \rho_w \rho'_r$ et $\rho_s = \rho_w \rho'_s$.
Par le lemme 4.5.14, nous avons soit $w \notin W$, soit $w \in W$ et $(w,r) \notin \mathcal{D}(U)$ ou bien $w \in W$ et $(w,s) \notin \mathcal{D}(V)$.
Cas $w \notin W$**.** Le couple (r,s) appartient donc à $\overline{N}_{First(\rho'_r)} \cdot T_{W^{\mathrm{c}}_{\mathrm{Last}(w)}} \cdot N_{\mathrm{First}(\rho'_s)}$.
Cas $w \in W$ **et** $(w,r) \notin \mathcal{D}(U)$ **.** Comme $(w,r) \notin \mathcal{D}(U)$, il existe $U' \in U^{\Subset}$ tel que $(w,r) \in \mathcal{D}(U'_{\mathrm{First}(\rho'_s)})$. Le couple (r,s) appartient donc à $\overline{U'}_{\mathrm{First}(\rho'_s)} \cdot T_{W_{\mathrm{Last}(w)}} \cdot N_{\mathrm{First}(\rho'_s)}$.
Cas $w \in W$ **et** $(w,r) \notin \mathcal{D}(V)$ **.** Ce cas est similaire au précédent.

Dans tous les cas, (r,s) appartient à $\mathcal{D}(C)$ et il suit donc que $\mathcal{D}(R)^{\mathrm{c}}$ est inclus dans $\mathcal{D}(C)$. □

4.5.3 Automates normalisés

Dans ce sous-paragraphe, nous présentons une notion d'accepteurs finis pour les langages de $\mathrm{Rat}_{k+1}(\Gamma)$ basée sur les relations de $\mathrm{PR}_k(\Gamma)$. Ces automates seront appelés automates normalisés de niveau $k+1$. Nous verrons que ces automates admettent une notion naturelle de déterminisme et de complétude et qu'ils permettent donc de réobtenir les propriétés de clôture de $\mathrm{Rat}_{k+1}(\Gamma)$. L'avantage de cette notion est que, contrairement à la notion d'automates réduits sur Γ_{k+1}, elle ne fait pas apparaître explicitement les suites réduites des piles de niveau $k+1$.

Intuitivement, au lieu de construire les piles de niveau $k+1$ en suivant leur suite réduite, ces automates les construisent la suite de leurs piles de niveau k de la première à la dernière.

Dans le théorème 4.3.50, nous avons établi que les langages de $\mathrm{Rat}_{k+1}(\Gamma)$ peuvent être décrits par des ensembles rationnels de suites d'instructions ne contenant pas l'instruction $\bar{k}$ si nous ajoutons les instructions de tests dans les langages de $\mathrm{Rat}_k(\Gamma)$. Nous avons pour tout niveau $k \geq 1$ que:

$$\mathrm{Rat}_{k+1}(\Gamma) = \bigcup_{\text{finie}} \mathrm{Rat}_k(\Gamma) \cdot \mathrm{Rat}(k \cdot \mathrm{Rat}(\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k(\Gamma)})^*)^*.$$

Par le corollaire $\mathcal{S}(\mathrm{Rat}(\Gamma_k \cup \mathcal{T}_{Rat_k(\Gamma)})^*) = \mathrm{Rat}_k(\Gamma)$,nous obtenons donc

$$\mathrm{Rat}_{k+1}(\Gamma) = \bigcup_{\text{finie}} \mathrm{Rat}((\mathrm{copy}_k \cdot \mathrm{PR}_k)^*)(\mathrm{Rat}_k(\Gamma)).$$

Nous en déduisons une notion d'automate fini étiqueté par $\mathrm{Rat}_k(\Gamma)$ et par $\mathrm{copy}_k \cdot \mathrm{PR}_k$ acceptant les langages de $\mathrm{Rat}_{k+1}(\Gamma)$. Au niveau 1, un automate normalisé est simplement un automate étiqueté par push_Γ.

Définition 4.5.17. Pour tout niveau $k \geq 1$, un automate A normalisé de niveau $k+1$ est un automate fini (Q,I,F,Δ) étiqueté par $\mathrm{Rat}_k(\Gamma) \cup (k \cdot \mathrm{Rew}_k(\Gamma))$ tel que:

- les états initiaux I n'apparaissent pas comme but d'une transition de Δ,
- pour toute transition $p \xrightarrow{R} q \in \Delta$, $R \in \mathrm{Rat}_k(\Gamma) \Leftrightarrow p \in I$.

Une pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ est acceptée par A s'il existe un calcul de l'automate A de la forme:

$$i \xrightarrow{R} p_1 \xrightarrow{\mathrm{copy}_k \cdot P_1} p_2 \ldots p_{n-1} \xrightarrow{\mathrm{copy}_k \cdot P_{n-1}} p_n$$

où $n \geq 1$, $i \in I$, $p_n \in F$, $i \xrightarrow{R} p_1 \in \Delta$ et pour tout $i \in [1,n-1]$, $p_i \xrightarrow{\mathrm{copy}_k \cdot P_i} p_{i+1} \in \Delta$ et que la pile s appartient à $(\mathrm{copy}_k \cdot P1 \cdot \ldots \mathrm{copy}_k P_{n-1})(R)$. Le langage de piles de niveau $k+1$ accepté par A sera noté $\mathcal{S}(A)$.

Exemple 4.5.18. Considérons l'automate normalisé de niveau 2 sur $\Gamma = \{a\}$ représenté dans la figure 4.9. Le langage de piles de niveau 2 accepté par A est l'ensemble:

$$\begin{aligned} \mathcal{S}(A) \;=\; \{[\,[\,a^{p_1}\,]\ldots[\,a^{p_n}\,]\,]_2 \mid\; & n \geq 2,\, p_1 > \cdots > p_n, \\ & \forall i \in [1,n-1], p_i \neq p_{i+1} \bmod 2 \text{ et } p_n = 0 \bmod 2\}. \end{aligned}$$

Remarque 4.5.19. Pour manipuler symboliquement un automate normalisé de niveau k, il nous faut fixer une représentation pour les ensembles et les relations apparaissant dans sa définition. Au niveau 1, nous représenterons naturellement

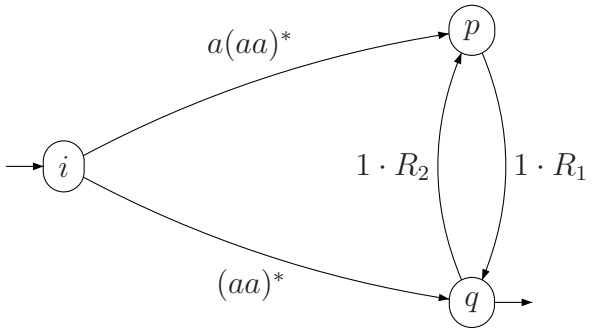


FIG. 4.9 – *Un automate normalisé de niveau* 2 *où* R_1 *et* R_2 *appartiennent à* Rew_1 *et sont respectivement égaux à* $\bar{a}^+ \cdot T_{(aa)^*}$ *et* $\bar{a}^+ \cdot T_{a(aa)^*}$.

---

un automate normalisé de niveau 1 par un automate fini étiqueté par Γ. Pour un automate normalisé de niveau $k \geq 2$, les ensembles de $\mathrm{Rat}_{k-1}(\Gamma)$ sont donnés par des automates réduits sur Γ_{k-1} avec tests dans $\mathcal{T}_{\mathrm{Rat}_{k-1}(\Gamma)}$ et les relations PR_{k-1} sont données par des unions finies d'éléments de Rew_{k-1} (cf. théorème 4.5.12). Un élément de Rew_{k-1} est représenté par un triplet (A,B,C) d'automates réduits sur Γ_{k-1} avec tests dans $\mathrm{Rat}_{k-1}(\Gamma)$ qui définit l'élément $\overline{\mathcal{L}(A)} \cdot T_{\mathcal{S}(B)} \cdot \mathcal{L}(C)$.

Il est bien entendu possible de donner une représentation symbolique des automates normalisés de niveau $k+1 \geq 2$ sans faire intervenir les automates réduits sur Γ_k avec tests dans $\mathrm{Rat}_k(\Gamma)$ en le remplaçant par des automates normalisés de niveau k. Il faut cependant adapter légèrement la définition des automates normalisés pour pouvoir décrire les ensembles de Norm_k apparaissant dans la définition des éléments de Rew_k. Pour cela, il suffit de considérer dans ce cas des automates normalisés non plus étiquetés par $\mathrm{Rat}_k(\Gamma)$ et $\mathrm{copy}_k \cdot \mathrm{PR}_k(\Gamma)$ mais par $\mathrm{PR}_k(\Gamma)$ et $\mathrm{copy}_k \cdot \mathrm{PR}_k(\Gamma)$. Nous n'adoptons toutefois pas cette représentation pour simplifier les preuves qui vont suivre.

Théorème 4.5.20. *Pour tout automate* A *sur* Γ_k*, il existe un automate normalisé* B *de niveau* k *acceptant* $\mathcal{S}(A)$*. De plus, la taille de* B *est bornée par* $\exp[k-1](|A|)$.

Démonstration. Au niveau 1, la propriété est immédiate. Considérons maintenant un automate sur Γ_k pour un niveau $k \geq 2$. Nous avons établi dans la proposition 4.3.48 et dans le théorème 4.4.15, qu'il existe un automate réduit B sur Γ_k avec des tests dans un ensemble fini $\mathcal{L} \subset \mathrm{Rat}_{k-1}(\Gamma)$ équivalent à A. De plus $|B|$ et $|\mathcal{L}|$ sont bornés par $\exp[0](|A|)$ et chaque langage $L \in \mathcal{L}$ est accepté par un automate A_L entièrement déterministe et complet sur Γ_{k-1} de taille bornée par $\exp[k-1](|A|)$.

En combinant les propositions 4.5.10 et 4.5.11, nous obtenons un automate

normalisé C de niveau k acceptant $\mathcal{S}(A)$ et de taille $\exp[k-1](|A|)$. □

Pour les automates normalisés de niveau 1, les notions de déterminisme et de complétude ont déjà été présentées. Nous dirons qu'un automate $A = (Q,I,F,\Delta)$ normalisé de niveau $k+1 \geq 2$ est dit *déterministe* si l'ensemble des états initiaux est réduit à un singleton et si pour tous états $p,q_1,q_2 \in Q$ avec $q_1 \neq q_2$, tout $N_1,N_2 \in \mathrm{Rat}_k(\Gamma)$ et pour tout $P_1,P_2 \in \mathrm{PR}_k(\Gamma)$,

$$\begin{cases} p \xrightarrow{N_1} q_1 \in \Delta \text{ et } p \xrightarrow{N_2} q_2 \in \Delta & \Rightarrow \quad R_1 \cap R_2 = \emptyset \\ p \xrightarrow{\mathrm{copy}_k \cdot P_1} q_1 \in \Delta \text{ et } p \xrightarrow{\mathrm{copy}_k \cdot P_2} q_2 \in \Delta & \Rightarrow \quad P_1 \cap P_2 = \emptyset. \end{cases}$$

De même, nous dirons que A est *complet* si pour tout $q \in Q \setminus I$ et $i \in I$,

$$\begin{cases} \bigcup_{i \xrightarrow{N} p \in \Delta} N & = & \mathrm{Stacks}_k(\Gamma), \\ \bigcup_{q \xrightarrow{\mathrm{copy}_k \cdot P} p \in \Delta} P & = & \mathrm{Stacks}_k(\Gamma) \times \mathrm{Stacks}_k(\Gamma). \end{cases}$$

En utilisant les propriétés de clôture des relations $\mathrm{PR}_k(\Gamma)$ et de $\mathrm{Rat}_k(\Gamma)$, nous obtenons par la méthode classique des *sous-ensembles* la déterminisation des automates normalisés.

Proposition 4.5.21. *Pour tout automate A normalisé de niveau k, il existe un automate B normalisé déterministe et complet de même niveau acceptant $\mathcal{S}(B)$.*

Démonstration. Au niveau 1, la propriété est déjà bien connue. Considérons un automate $A = (Q_A,I_A,F_A,\Delta_A)$ normalisé de niveau $k+1 \geq 2$. Comme les relations de PR_k sont closes par union, nous pouvons sans perte de généralité supposer que pour tous états p et $q \in Q_A$, il existe au plus une transition $p \xrightarrow{N_{p,q}} q$ ou $p \xrightarrow{\mathrm{copy}_k \cdot P_{p,q}} q$ dans Δ_A. S'il n'existe pas de telle transition, nous prendrons $N_{p,q} = \emptyset$ et $P_{p,q} = \emptyset$.

Nous construisons un automate $B = (Q_B,I_B,F_B,\Delta_B)$ normalisé et déterministe de niveau k acceptant $\mathcal{S}(A)$. L'ensemble des états Q_B est égal à 2^{Q_A}. L'ensemble des états initiaux I_B est réduit au singleton $\{ I_A \}$. L'ensemble des états finaux F_B est égal à $\{Q \subseteq Q_A \mid Q \cap F_A = \emptyset\}$. Enfin, l'ensemble des transitions Δ_B est défini ci-dessous.
Pour tout $Q \subseteq Q_A$, une transition de la forme:

$$I_A \xrightarrow{N_Q} Q \in \Delta$$

où $N_Q = \bigcap_{p \in Q} (\bigcup_{i \in I_A} N_{i,p}) \cap \bigcup_{p \in (Q_A \setminus Q)} (\bigcap_{i \in I_A} (N_{i,p})^{\mathrm{c}})$.
Pour tout $Q,Q' \subseteq Q_A$, une transition de la forme:

$$Q \xrightarrow{\mathrm{copy}_k \cdot N_{Q,Q'}} Q' \in \Delta$$

où $N_{Q,Q'} = \bigcap_{q \in Q} (\bigcup_{q' \in Q'} P_{q,q'}) \cap \bigcup_{q \in (Q_A \setminus Q)} (\bigcap_{q' \in Q'} (P_{q,q'})^{\mathrm{c}})$.

Comme $\mathrm{PR}_k(\Gamma)$ et $\mathrm{Rat}_k(\Gamma)$ sont des algèbre de Boole, B est bien un automate normalisé. De plus par construction, B est déterministe et complet. On vérifie aisément que $\mathcal{S}(B) = \mathcal{S}(A)$. □

Les automates normalisés de niveau k permettent de réobtenir au moyen de constructions naturelles les propriétés de clôture des langages de $\mathrm{Rat}_k(\Gamma)$. En particulier, le complémentaire du langage accepté par un automate normalisé déterministe et complet est obtenu en prenant le complémentaire de l'ensemble des états finaux.

Nous concluons ce paragraphe en donnant la complexité de la transformation d'un automate sur Γ_k en un automate normalisé déterministe et complet de niveau k.

Théorème 4.5.22. *Pour tout automate A sur Γ_k, il existe un automate normalisé B déterministe et complet de niveau k acceptant $\mathcal{S}(A)$. De plus, la taille de l'automate B est bornée par* $\exp[k](|A|)$.

4.6 Rationalité induite par COps_k.

Dans ce paragraphe, nous comparons la notion de rationalité induite par le jeu d'opérations Ops_k à celle induite par le jeu d'opérations classiques COps_k. Nous définissons ainsi:

$$\mathrm{CRat}_k(\Gamma) = \mathrm{Rat}(\mathrm{COps}_k^\star)([\ ]_k) = \mathcal{R}^{\mathrm{c}}(\mathrm{Rat}(\Gamma_k^*))([\ ]_k).$$

Nous ne nous intéressons pas aux aspects algorithmiques mais seulement à la comparaison de l'expressivité des deux notions. Le but de ce paragraphe est de convaincre le lecteur du manque de pertinence de la rationalité induite par COps_k et de justifier *a posteriori* l'emploi du jeu d'opérations symétriques Ops_k.

Remarquons que quelque soit le niveau $k \geq 1$, l'ensemble $\mathrm{CRat}_k(\Gamma)$ est inclus dans l'ensemble $\mathrm{Rat}_k(\Gamma)$. En effet, comme nous l'avons déjà remarqué pour tout $\ell \geq 1$, nous avons:

$$\mathrm{destr}_\ell = \mathrm{Ops}_\ell^\star \cdot \overline{\mathrm{copy}}_\ell.$$

Aux niveaux 1 et 2, l'inclusion réciproque tient et les deux notions de rationalité coïncident.

Proposition 4.6.1. *Pour tout alphabet fini Γ,*

$$\mathit{Rat}_1(\Gamma) = \mathit{CRat}_1(\Gamma) \qquad \mathit{Rat}_2(\Gamma) = \mathit{CRat}_2(\Gamma).$$

Démonstration. Au niveau 1, l'égalité est immédiate car $\mathrm{Ops}_1(\Gamma) = \mathrm{COps}_1(\Gamma)$.

Au niveau 2, il nous suffit de montrer que $\mathrm{Rat}_2(\Gamma)$ est inclus dans $\mathrm{CRat}_2(\Gamma)$. Nous avons établi dans la proposition 4.3.49 que:

$$\mathrm{Rat}_2(\Gamma) = \mathcal{R}\left(\mathrm{Rat}\left(\left(\Gamma_1 \cup \{1, \bot_2\} \cup \mathcal{T}_{\mathrm{Rat}_1(\Gamma)}\right)^*\right)\right)([\,]_2).$$

Or, pour tout $R \in \mathrm{Rat}_1(\Gamma)$, l'opération de test Test^2_R est égale à $\mathcal{R}^{\mathrm{c}}(1 \cdot \overline{R} \cdot \bot_1 \cdot \bar{1})$. Il en découle que $\mathrm{Rat}_2(\Gamma) \subseteq \mathrm{CRat}_2(\Gamma)$. □

Au niveau 3, nous verrons que $\mathrm{CRat}_3(\Gamma)$ est strictement inclus dans $\mathrm{Rat}_3(\Gamma)$ et que de plus $\mathrm{CRat}_3(\Gamma)$ n'est pas une algèbre de Boole. Considérons le langage S de piles de niveau 3 sur $\Gamma = \{a\}$ défini par:

$$S = \{[\,[\,[\,a^{p_1}\,]\ldots[\,a^{p_m}\,]\,]_2\,[\,[\,a^{p_1}\,]\ldots[\,a^{p_n}\,]\,]_2\,]_3 \mid n \leq m \text{ et } p_n = p_{n-1}\}.$$

Le langage S appartient à $\mathrm{Rat}_3(\Gamma)$ car $S = \mathcal{R}\left((\Gamma_1^* \cdot 1)^* \cdot 2 \cdot (\Gamma_1^* \cdot \bar{1})^* \cdot \bar{1}\right)([\,]_3)$. La suite de ce paragraphe est consacrée à établir que S n'appartient pas à $\mathrm{CRat}_3(\Gamma)$. Admettons momentanément ce résultat. Le langage S peut s'exprimer comme l'intersection des deux langages S_1 et S_2 donnés ci-dessous:

$$\begin{array}{rcl} S_1 & = & \{[\,[\,[\,a^{p_1}\,]\ldots[\,a^{p_m}\,]\,]_2\,[\,[\,a^{p_1}\,]\ldots[\,a^{p_n}\,]\,]_2\,]_3 \mid n \leq m\} \\ S_2 & = & \{\left[\,[\,[\,a^{p_1}\,]\ldots[\,a^{p_m}\,]\,]_2\,\left[\,\left[\,a^{p'_1}\,\right]\ldots\left[\,a^{p'_n}\,\right]\,\right]_2\,\right]_3 \mid n \geq 2 \text{ et } p'_{n-1} = p'_n\} \end{array}$$

Ces deux ensembles appartiennent à $\mathrm{CRat}_3(\Gamma)$. En effet,

$$\begin{array}{rcl} S_1 & = & \mathcal{R}^C((\Gamma_1^* \cdot 1)^* \cdot 2 \cdot \bar{1}^*)([\,]_3) \\ S_2 & = & \mathcal{R}^C\left((\Gamma_1^* \cdot 1)^* \cdot 2\bar{1}^*\overline{\Gamma}^* \cdot \bot_2 \cdot (\Gamma_1^* \cdot 1)^* \cdot 1\right)([\,]_3). \end{array}$$

L'ensemble $\mathrm{CRat}_3(\Gamma)$ n'est donc pas clos par intersection. L'utilisation du jeu d'opérations classiques fait donc perdre la fermeture par complémentaire.

Pour pouvoir établir que le langage S n'appartient pas à $\mathrm{CRat}_3(\Gamma)$, il nous faut tout d'abord donner une représentation normalisée de ces ensembles. La première étape est d'établir que les ensembles de $\mathrm{CRat}_3(\Gamma)$ peuvent être décrits sans utiliser l'opération destr_2 si l'on ajoute des langages de tests. Comme dans le cas du jeu d'opérations symétriques, ces langages de tests sont définis au moyen d'automates alternants de niveau 2 sur les opérations classiques. La différence vient du fait que sur le jeu d'opérations classiques, les automates alternants sont moins expressifs que les automates non-alternants.

4.6.1 Automates alternants sur Γ_1^{c} et Γ_2^{c}.

Ce sous-paragraphe étudie l'expressivité des automates alternants de niveau 1 et 2 sur le jeu d'opérations classiques. Un automate alternant de niveau k sur le jeu d'opérations classiques a la même définition formelle qu'un automate

alternant sur Γ_k (cf. définition 4.3.10). Cependant, la notion d'exécution diffère. Elle est obtenue en remplaçant $\mathcal{R}$ par $\mathcal{R}^{\mathrm{c}}$ dans la définition de l'exécution d'un automate alternant sur Γ_k. Dans la suite, nous parlerons d'automate alternant sur Γ_k^{c} pour désigner les automates alternants de niveau k sur le jeu d'opérations classiques $\mathrm{COps}_k(\Gamma)$ et nous noterons $\mathrm{CAlt}_k(\Gamma)$ l'ensemble des langages acceptés par les automates alternants sur Γ_k^{c}.

Un premier résultat immédiat est que ces nouveaux automates peuvent être simulés par les automates alternants sur Γ_k.

Proposition 4.6.2. *Pour tout $k \geq 1$ et tout alphabet fini Γ,*

$$CAlt_k(\Gamma) \subseteq Alt_k(\Gamma).$$

Démonstration. Comme nous l'avons déjà remarqué, nous avons pour tout $\ell \geq 1$ l'égalité suivante:

$$\mathrm{destr}_\ell = \mathrm{Ops}_\ell^\star \cdot \overline{\mathrm{copy}}_\ell.$$

Il est donc immédiat de construire pour tout automate alternant sur Γ_k^{c}, un automate équivalent sur Γ_k. □

Au niveau 1, les deux modèles d'automates ont la même expressivité.

Proposition 4.6.3. *Les langages acceptés par les automates alternants sur Γ_1^{c} sont les langages de Rat_1.*

Démonstration. Comme les ensembles $\mathrm{Ops}_1(\Gamma)$ et $\mathrm{COps}_1(\Gamma)$ sont égaux, les ensembles acceptés par les automates alternants sur Γ_1 et Γ_1^{c} coïncident. La propriété annoncée suit donc par la proposition 4.3.31. □

Dès le niveau 2, les automates alternants sur Γ_2^{c} ne sont pas assez puissants pour capturer tous les langages de $\mathrm{Rat}_2(\Gamma)$. En effet, du fait de l'absence de symétrie entre copy_1 et destr_1, une construction similaire à celle de la proposition 4.3.14 ne peut être utilisée.

Les langages acceptés par les automates alternants sur Γ_2^{c} sont aisément caractérisés en introduisant l'opération new_1 qui empile une pile vide de niveau 1 sur la dernière pile de niveau 2 (*i.e.* pour toute pile $s = [\, s_1, \ldots, s_n \,]_2$, $\mathrm{new}_1(s) = [\, s_1, \ldots, s_n, [\,]\,]_2$). Nous noterons N_1 l'instruction correspondante. Nous définissons les ensembles rationnels faibles comme les ensembles obtenus en remplaçant l'opération copy_1 par l'opération new_1 dans la définition de $\mathrm{CRat}_2(\Gamma)$. L'ensemble de tous les ensembles rationnels faibles de niveau 2 est noté $\mathrm{WRat}_2(\Gamma)$ et est défini par:

$$\begin{aligned} \mathrm{WRat}_2(\Gamma) &= \mathrm{Rat}\left((\mathrm{Ops}_1 \cup \{\, \mathrm{new}_1 \,\})^*\right)([\,]_2) \\ &= \mathrm{Rat}\left((\mathrm{Ops}_1 \cup \{\, \mathrm{new}_1, \mathrm{destr}_1 \,\})^*\right)([\,]_2). \end{aligned}$$

Remarque 4.6.4. Ces langages ont été définis à tout niveau dans [BM04]. Les auteurs passent pas la représentation des piles de piles sur Γ par des mots sur l'alphabet $\Gamma \cup \{\,[\,,]\,\}$. Ainsi, à chaque pile $s = [\,s_1,\ldots,s_n\,]_2$ de $\mathrm{Stacks}_2(\Gamma)$, est associé le mot $\Phi(s) = [[s_1]\ldots[s_n]]$ sur l'alphabet $\Gamma \cup \{\,[\,,]\,\}$. Il est aisé de vérifier qu'un ensemble $R \subseteq \mathrm{Stacks}_2(\Gamma)$ est faiblement rationnel si et seulement si $\Phi(R)$ est un ensemble rationnel de mots. Il s'en suit donc que l'ensemble $\mathrm{WRat}_2(\Gamma)$ est une algèbre de Boole.

Comme $\mathrm{new}_1 = \mathrm{copy}_1 \cdot \mathrm{pop}_\Gamma^* \cdot T_{[\,]_1}$, $\mathrm{WRat}_2(\Gamma)$ est inclus dans $\mathrm{Rat}_2(\Gamma)$, cette inclusion est stricte comme le montre l'exemple suivant.

Exemple 4.6.5. L'ensemble L_1 de piles de niveau 2 sur l'alphabet $\Gamma = \{\,a,b\,\}$ défini comme l'ensemble $\{[\,[\,a^{2n_1}\,]\,[\,b^{2m_1+1}\,]\,,\ldots,[\,a^{2n_p}\,]\,[\,b^{2m_p+1}\,]\,]_2 \mid p \geq 1\}$ appartient à $\mathrm{WRat}_2(\Gamma)$ car $\Phi(L_1)$ est égal à $[([(aa)^*][b(bb)^*])^+]$. Le langage S appartient bien à $\mathrm{Rat}_2(\Gamma)$ car il est égal à:

$$\mathcal{R}\left((aa)^*1\bar{a}^*\perp_1 b(bb)^*\left(1\bar{b}^*\perp_1(aa)^*1\bar{a}^*\perp_1 b(bb)^*\right)^*\right)([\,]_2).$$

L'ensemble L_2 de piles de niveau 2 sur l'alphabet $\Gamma = \{\,a\,\}$ défini comme l'ensemble $\{[\,[\,a^n\,]\,[\,a^n\,]\,]_2 \mid n \geq 1\}$ appartient à $\mathrm{Rat}_2(\Gamma)$ car $L_2 = \mathcal{R}(a^*1)([\,]_2)$. Le langage L_2 n'est pas faiblement rationnel car l'ensemble des mots $\Phi(L_2)$ n'est pas un langage rationnel.

La proposition suivante établit que les langages acceptés par les automates alternants sur Γ_2^{c} sont les langages rationnels faibles de niveau 2.

Proposition 4.6.6. *Les langages acceptés par les automates alternants sur* Γ_2^{c} *sont les langages de* $WRat_2(\Gamma)$.

Démonstration. L'inclusion réciproque est immédiate. Il suffit donc de montrer l'inclusion directe. Soit $A = (Q_A,I_A,\Delta_A)$ un automate alternant sur Γ_2^{c}. Nous allons montrer que $\mathcal{S}(A)$ est un ensemble rationnel faible.

Pour toute pile $s \in \mathrm{Stacks}_2(\Gamma)$, nous noterons X_s l'ensemble des états initiant un calcul de A partant de s (*i.e.* $X_s = \{q \in Q_A \mid s \in \mathcal{S}_q(A)\}$).

Fait 1. Pour toutes piles s et s' dans $\mathrm{Stacks}_2(\Gamma)$ telles que $X_s = X'_s$ et pour toute pile r dans $\mathrm{Stacks}_1(\Gamma)$, les piles $s \cdot r$ et $s' \cdot r$ satisfont $X_{s\cdot r} = X_{s'\cdot r}$.

Pour chaque exécution $\mathcal{E} = (T,C)$ de A partant de $s \cdot r$ dans l'état p, nous construisons une exécution $\mathcal{E}' = (T',C')$ de A partant de $s' \cdot r$ dans l'état p. L'exécution $\mathcal{E}'$ est obtenue en substituant aux sous-exécutions de $\mathcal{E}$ commençant par p_0 en s une exécution de A commençant par p_0 en s' (cette exécution existe car $X_s \subseteq X'_s$) puis en remplaçant $s\cdot t$ par $s'\cdot t$ (quel que soit la pile $t \in \mathrm{Stacks}_1(\Gamma)$) dans l'image par C' des noeuds de T' qui n'ont pas été introduits par la substitution. Nous déduisons donc que $X_{s\cdot r} \subseteq X_{s\cdot r'}$ et par symétrie, nous obtenons l'égalité.

Nous associons à chaque pile r de niveau 1, une fonction partielle de 2^{Q_A} dans 2^{Q_A} notée f_r et définie par:

$$f_r(P) = \begin{cases} X_{s\cdot r} & \text{s'il existe } s \in \mathrm{Stacks}_2(\Gamma) \text{ telle que } X_s = P, \\ \text{non définie} & \text{sinon.} \end{cases}$$

Pour toute fonction partielle f de 2^{Q_A} dans 2^{Q_A}, nous noterons $X_f = \{r \in \mathrm{Stacks}_1(\Gamma) \mid f = f_r\}$.

Fait 2. Pour toute fonction partielle f de 2^{Q_A} dans 2^{Q_A}, l'ensemble X_f est un ensemble rationnel.

Soit f une fonction partielle de 2^{Q_A} dans 2^{Q_A}. Nous pouvons supposer, sans perte de généralité, que $\mathrm{Dom}(f) = \{P \mid \exists s \in \mathrm{Stacks}_2(\Gamma), X_s = O\}$. En effet, si ce n'est pas le cas alors X_f est égal à l'ensemble vide. Pour tout $P \in \mathrm{Dom}(f)$, s_P désignera une pile de niveau 2 telle que $X_{s_P} = P$.

Il nous suffit d'établir, pour tout $P \in \mathrm{Dom}(f)$ et pour tout $p \in Q_A$, que l'ensemble $S_{P,p} = \{r \in \mathrm{Stacks}_1(\Gamma) \mid s_P \cdot r \in \mathcal{S}_p(A)\}$ est rationnel. Pour cela, il suffit de remarquer que:

$$S_{P,p} = \mathrm{top}_1\left(\mathcal{S}_p(A) \cap \{s_P \cdot r \mid r \in \mathrm{Stacks}_1(\Gamma)\}\right).$$

En effet comme $\mathrm{CAlt}_2 \subseteq \mathrm{Alt}_2 = \mathrm{Rat}_2$, le langage $\mathcal{S}_p(A)$ appartient à Rat_2. Comme le langage $\{s_p \cdot r \mid r \in \mathrm{Stacks}_1(\Gamma)\}$ est égal à $\mathcal{R}(\rho_{s_P} 1\Gamma_1^*)([\,]_2)$, il appartient à $\mathrm{Rat}_2(\Gamma)$. Il suit donc par le théorème 4.4.8 que $\mathcal{S}_p(A) \cap \{s_p \cdot r \mid r \in \mathrm{Stacks}_1(\Gamma)\}$ appartient à Rat_2. Par le corollaire 4.3.51, le langage $S_{P,p}$ appartient à $\mathrm{Rat}_1(\Gamma)$.

Par un raisonnement similaire, nous établissons le fait suivant.

Fait 3. Pour tout ensemble $P \subseteq Q_A$, l'ensemble des piles de niveau 1, $E_P = \{r \in \mathrm{Stacks}_1(\Gamma) \mid X_{[r]_2} = P\}$ appartient à Rat_1.

Considérons l'ensemble R de suites d'instructions de $\Gamma \cup \{\mathrm{N}_1\}$ égal à:

$$\bigcup_{P_0 \subseteq Q_A} \{E_{P_O} \mathrm{N}_1 X_{f_1}, \ldots, X_{f_{\ell-1}} \mathrm{N}_1 X_{f_\ell} \mid \ell \geq 0 \text{ et } (f_1 \cdots f_\ell)(P_0) \cap I_A \neq \emptyset\}.$$

D'après le fait 1, le langage accepté par A est égal à $\mathcal{R}(R)([\,]_2)$. D'après les faits 2 et 3, R appartient à $\mathrm{Rat}((\Gamma \cup \{\mathrm{N}_1\})^*)$. Donc $\mathcal{S}(A)$ appartient à $\mathrm{WRat}_2(\Gamma)$. □

Remarque 4.6.7. Cette propriété s'étend à tout niveau k et les langages acceptés par les automates alternants sur Γ_k^c sont les langages de $\mathrm{WRat}_k(\Gamma)$.

Pour des raisons techniques, nous adaptons la notion d'automate normalisé, présentée dans le sous-paragraphe 4.5.3 pour les ensembles de WRat_2. La notion d'*automate normalisé faible de niveau* 2 est obtenue en remplaçant $1 \cdot \mathrm{Rew}_1$ par $\mathrm{N}_1 \cdot \mathrm{Rat}_1$ dans la définition 4.5.17. Les notions de déterminisme et de complétude

sont définies de manière analogue. Par une adaptation immédiate de la preuve de la proposition 4.5.21, nous établissons que les automates normalisés faibles de niveau 2 peuvent être déterminisés.

Proposition 4.6.8. *Tout langage de $WRat_2(\Gamma)$ est accepté par un automate normalisé faible de niveau 2 déterministe et complet.*

4.6.2 Représentation normalisée de $\mathbf{CRat}_3(\Gamma)$.

La première étape de normalisation consiste à donner une description des ensembles de $\mathrm{CRat}_3(\Gamma)$ qui ne fasse pas intervenir l'opération destr_2. Comme dans le cas du jeu d'opérations symétriques, il est nécessaire d'introduire des tests acceptés par des automates alternants sur Γ_2^{c} (qui, d'après la proposition 4.6.6, sont les ensembles faiblement rationnels de niveau 2).

Proposition 4.6.9. *Pour tout alphabet fini Γ,*

$$CRat_3 = \mathcal{R}^{\mathrm{c}}(Rat((\Gamma_2 \cup \{\,2\,\} \cup \mathcal{T}_{WRat_2})^*))([\,]_2).$$

Démonstration. Pour l'inclusion réciproque, il suffit de remarquer que pour tout $R \in \mathrm{WRat}_2(\Gamma)$, il existe Test_R^3 appartenant à $\mathcal{R}^{\mathrm{c}}(2 \cdot \mathrm{Rat}(\Gamma_2^*) \cdot \bar{2})$.

Pour l'inclusion directe, nous adaptons l'approche développée dans le sous-paragraphe 4.3.3.2. Soit $A = (Q_A, I_A, F_A, \Delta_A)$ un automate sur Γ_3^{c} n'ayant pas ε-transitions. Nous adaptons la notion de boucle au cas des opérations classiques. Pour tout $p,q \in Q_A$, nous définissons l'ensemble $R_{p,q}$ des piles s non-vides de niveau 3 telles que A boucle sur s en partant de p et en revenant en q. Une pile non-vide $s \in \mathrm{Stacks}_3(\Gamma)$ appartient à $R_{p,q}$ s'il existe un calcul de l'automate A:

$$(p,s) \xrightarrow[A]{2} (p_0,s_0) \xrightarrow[A]{\gamma_1} \cdots \xrightarrow[A]{\gamma_n} (p_n,s_n) \xrightarrow[A]{\bar{2}} (q,s)$$

tel que $n \geq 0$ et tel que pour tout $i \in [0,n]$, la pile s_i est soit égale à s soit de la forme $s \cdot s_i'$ pour une certaine pile $s_i' \in \mathrm{Stacks}_3(\Gamma)$.

En adaptant la preuve du lemme 4.3.46, nous construisons, pour tout $p_0,q_0 \in Q_A$, un automate A_{p_0,q_0} alternant sur Γ_2^{c} tel que pour toute pile non-vide s de niveau 3, $s \in R_{p_0,q_0}$ si et seulement si $\mathrm{top}_2(s) \in \mathcal{S}(A_{p_0,q_0})$. Comme R_{p_0,q_0} ne contient pas la pile vide $[\,]_3$, aucune pile n'intervenant dans les calculs définissant les boucles n'est vide, et donc nous pouvons sans perte de généralité supposer que A ne contient pas d'occurrence de l'instruction $\perp_3$.

Soient p_0,q_0 deux états de Q_A. Nous construisons l'automate $B = (Q_B, I_B, \Delta_B)$ alternant sur Γ_2^{c} en prenant $Q_B = Q_A \times Q_A \times \Gamma_2^{\mathrm{T}} \cup \{\,\bullet\,\}$ et $I_B = \{\,\bullet\,\}$. Nous définissons maintenant l'ensemble des transitions.

Pour tout $T \in \Gamma_2^{\mathrm{T}}$ et pour tout $R \subseteq Q_A \times Q_A$ tel que $(p_0,q_0) \in R^*$,

$$\bullet,T \to \bigwedge_{(p,q)\in R} ((p,q,T_{p,q}),\varepsilon) \in \Delta_B$$

si pour tout $(p,q) \in R$, il existe deux transitions $\delta_1 = p_0 \xrightarrow{2} p,T_1$ et $\delta_2 = q \xrightarrow{\bar{2}} q_0,T_2$ dans Δ_A avec $T_1 \leq T_{(p,q)}$ et $T_2 \leq T$.

Pour tout $(p_1,q_1,T) \in Q_A$, pour tout $R \subseteq Q_A \times Q_A$ et pour tout $\gamma \in \Gamma_2^* \cup \{\varepsilon\}$,

$$(p_1,q_1,T),T \to \bigwedge_{(p,q)\in R} ((p,q,T_{p,q}),\varepsilon) \ \wedge \ ((p_3,q_1,T'),\gamma) \in \Delta_B$$

s'il existe p_2 et $p_3 \in Q_A$ tels que $(p_1,p_2) \in R^*$ et tels que:

- pour tout $(p,q) \in R$, il existe deux transitions $\delta_1 = p_1 \xrightarrow{2} p,T_1$ et $\delta_2 = q \xrightarrow{\bar{2}} q_1,T_2$ dans Δ_A avec $T_1 \leq T_{(p,q)}$ et $T_2 \leq T$,
- $\gamma \in \Gamma_2^*$ et si $\gamma = \varepsilon$ alors $p_2 = p_3$ et sinon il existe une transition $p_2 \xrightarrow{\gamma} p_3,T''$ dans Δ_A avec $T'' \leq T'$.

Pour tout $p \in Q_A$ et $T \in \Gamma_2^{\mathrm{T}}$,

$$(p,p,T),T \to \emptyset \in \Delta_B.$$

Par construction et pour toute pile non-vide $s \in \mathrm{Stacks}_3(\Gamma)$, $s \in R_{p_0,q_0}$ si et seulement si $\mathrm{top}_2(s) \in \mathcal{S}(B)$.

En adaptant la construction de la proposition 4.3.48 et d'après la proposition 4.6.6, l'égalité annoncée en découle. □

L'étape suivante est de normaliser les relations induites par les ensembles d'opérations dans $\mathcal{R}^{\mathrm{c}}((\Gamma_2 \cup \mathcal{T}_{\mathrm{WRat}_2})^*)$. Il est possible de définir une forme normale Rew_2 analogue à celle de Rew_2. Nous définissons à cet effet:

$$\mathrm{Rew}_2^{\mathrm{c}} = \mathrm{Rat}((\mathcal{T}_{\mathrm{Rat}_1} \cdot \bar{1})^*) \cdot \mathcal{T}_{\mathrm{WRat}_2} \cdot \mathrm{Rew}_1 \cdot \mathrm{Rat}((1 \cdot \mathrm{Rew}_1)^*).$$

Proposition 4.6.10. *Toute relation de* $\mathcal{D}^{\mathrm{c}}(Rat((\Gamma_2 \cup \mathcal{T}_{WRat_2})^*))$ *peut s'écrire comme une union finie de relations dans* $\mathcal{D}^{\mathrm{c}}(Rew_2^{\mathrm{c}})$.

Démonstration. La preuve de cette égalité se décompose en plusieurs étapes.
Fait 1. Toute relation de $\mathcal{D}^{\mathrm{c}}(\mathrm{Rat}((\Gamma_2 \cup \mathcal{T}_{\mathrm{WRat}_2})^*))$ est une union finie de relations dans

$$\mathcal{D}^{\mathrm{c}}\left(\mathrm{Rat}\left((\mathrm{WR}_1 \cdot \bar{1})^*\right) \cdot \mathrm{WR}_1 \cdot \mathrm{Rat}\left((1 \cdot \mathrm{WR}_1)^*\right)\right)$$

où WR_1 désigne l'ensemble $\mathrm{Rat}((\Gamma_1 \cup \mathcal{T}_{\mathrm{WRat}_2})^*)$.

Cette première étape est une adaptation de la preuve de la proposition 4.5.11. Soit A un automate sur Γ_2^{c} avec tests dans $\mathcal{L} = \{L_1,\ldots,L_n\} \subseteq \mathcal{T}_{\mathrm{WRat}_2}$. Pour

tout $\ell \in [1,n]$, nous notons $A_i = (Q_i,\{\, i_\ell \,\},F_\ell,\Delta_\ell)$ un automate normalisé faible de niveau 2, déterministe et complet acceptant L_ℓ. Comme A_ℓ est déterministe et complet, il existe pour toute pile $s \in \mathrm{Stacks}_2$ un unique état $q_s \in Q_\ell$ tel qu'il existe un calcul de A_ℓ partant de $[\,]_2$ dans l'état initial i_ℓ et arrivant en s dans l'état q_s. Nous noterons T l'ensemble de tous les langages de $\mathrm{Rat}_1(\Gamma)$ étiquetant les automates A_ℓ. Nous désignerons cet unique état par $A_\ell(s)$. A chaque pile $s \in \mathrm{Stacks}_2(\Gamma)$, nous associons le uplet $W_s = (A_1(s),\ldots,A_n(s))$.

Pour tout $d \in \prod_{\ell\in[1,n]} Q_\ell$ et pour tout $\ell \in [1,n]$, il existe un ensemble T_d^ℓ de parties de T tel que pour toute piles $s \in \mathrm{Stacks}_2(\Gamma)$ avec $W_s = c$ et pour toute pile $r \in \mathrm{Stacks}_1(\Gamma)$,

$$s \cdot r \in L_\ell \;\Leftrightarrow\; r \in \bigcup_{R\in T_d^\ell} \bigcap_{L\in R} L. \tag{4.10}$$

Comme dans la proposition 4.6.9, nous définissons les langages de boucles de l'automate A. Pour tout $p,q \in Q_A$, nous définissons l'ensemble $R_{p,q}$ des piles s de niveau 2 telles que A boucle sur s en partant en p et en revenant en q. Une pile $s \in \mathrm{Stacks}_3(\Gamma)$ appartient à $R_{p,q}$ s'il existe un calcul de l'automate A:

$$(p,s) \xrightarrow[A]{1} (p_0,s_0) \xrightarrow[A]{\gamma_1} \cdots \xrightarrow[A]{\gamma_n} (p_n,s_n) \xrightarrow[A]{\bar{1}} (q,s)$$

tel que $n \geq 0$ et tel que pour tout pour $i \in [0,n]$, la pile s_i est soit égale à s soit de la forme $s \cdot s'_i$ pour une certaine pile $s'_i \in \mathrm{Stacks}_3(\Gamma)$.

Comme A contient des tests dans WRat_2, l'appartenance d'une pile $s \in \mathrm{Stacks}_2(\Gamma)$ dépend de $\mathrm{top}_1(s)$ et W_s. Plus précisément, pour tout $p_0,q_0 \in Q_A$ et pour tout $t \in \prod_{\ell\in[1,n]} Q_\ell$, il existe un automate alternant $A^t_{p_0,q_0}$ sur Γ_1^{c} avec tests dans $T \subseteq \mathrm{Rat}_1(\Gamma)$ tel que pour toute pile $s \in \mathrm{Stacks}_2(\Gamma)$ vérifiant $W_s = c$, $s \in R_{p_0,q_0}$ si et seulement si $\mathrm{top}_1(s) \in \mathcal{S}(A^c_{p_0,q_0})$.

La construction de $A^{\mathrm{c}}_{p_0,q_0}$ est une adaptation immédiate de la construction de A_{p_0,q_0} dans la proposition 4.6.9 en utilisant l'équation 4.10. Par la proposition 4.3.38, $\mathcal{S}(A^c_{p_0,q_0})$ appartient à $\mathrm{Rat}_1(\Gamma)$ et donc à $\mathrm{WRat}_2(\Gamma)$.

En adaptant la construction de la proposition 4.5.9, nous obtenons le résultat annoncé.

Fait 2. Toute relation de $\mathcal{D}^{\mathrm{c}}(\mathrm{WR}_1\mathrm{Rat}((1\mathrm{WR}_1)^*))$ est une union finie de relations de la forme $\mathcal{D}^{\mathrm{c}}(\mathcal{T}_{\mathrm{WRat}_2} \cdot \mathrm{R}_1\mathrm{Rat}((1\mathrm{R}_1)^*))$ où R_1 désigne l'ensemble $\mathrm{Rat}((\Gamma_1 \cup \mathcal{T}_{\mathrm{Rat}_1})^*)$.
Toute relation de $\mathcal{D}^{\mathrm{c}}(\mathrm{Rat}\left((\mathrm{WR}_1 \cdot \bar{1})^*\right))$ est une union finie de relations dans $\mathcal{D}^{\mathrm{c}}(\mathrm{Rat}\left((\mathrm{R}_1 \cdot \bar{1})^*\right)) \cdot \mathcal{T}_{\mathrm{WRat}_2}$ où R_1 désigne l'ensemble $\mathrm{Rat}((\Gamma_1 \cup \mathcal{T}_{\mathrm{Rat}_1})^*)$.

Nous établissons la première propriété. La preuve de la deuxième propriété est similaire. Soit A un automate fini étiqueté par $\Gamma_1 \cup \{\,1\,\} \cup \mathcal{T}_{\mathrm{WRat}_2}$ et soit $\mathcal{L} = \{\, L_1,\ldots,L_n \,\} \subset \mathrm{WRat}_2$ l'ensemble des langages apparaissant dans une instruction

de test étiquetant A. Pour tout $\ell \in [1,n]$, soit $A_\ell = (Q_\ell,\{\, i_\ell \,\},F_\ell,\Delta_\ell)$ un automate normalisé faible de niveau 2 déterministe et complet.

À chaque pile $s = [\, s_1,\ldots,s_n \,]_2$ de niveau 2, nous associons un uplet $d_s = (d_1,\ldots,d_n)$ dans $D = \prod_{\ell\in[1,n]} Q_\ell \cup \prod_{\ell\in[1,n]}\{\, \bullet \,\}$ défini par:

$$\begin{cases} d_1 = \ldots = d_n = \bullet & \text{si } n = 1, \\ d_\ell = A_\ell([\, s_1,\ldots,s_{n-1} \,]_2), \text{pour tout } \ell \in [1,n] & \text{si } n > 1. \end{cases}$$

Avant de continuer, nous établissons quelques propriétés liées à cette notion.

Pour tout $d \in D$, l'ensemble $S_d = \{s \in \mathrm{Stacks}_2(\Gamma) \mid d_s = d\}$ des piles de niveau 2 est un ensemble rationnel faible.

Pour tout $d \in D$ et pour tout $\ell \in [1,n]$, il existe un ensemble T_d^ℓ de Rat_1 tel que pour toute pile $s \in \mathrm{Stacks}_2(\Gamma)$ avec $d_s = d$, $s \in L_\ell$ si et seulement si $\mathrm{top}_1(s) \in T_d^\ell$. Il suffit de prendre T_d^ℓ égal à l'union des R tel que $i_\ell \xrightarrow[A_\ell]{R} f$ avec $f \in F_\ell$ si $d = (\bullet,\ldots,\bullet)$ ou égal à l'union des R tels que $d_\ell \xrightarrow[A_\ell]{1\cdot R} f$ avec $f \in F_\ell$ sinon.

Pour tout $d,d' \in D$ et pour tout $\ell \in [1,n]$, il existe un ensemble $T_{d,d'}$ de Rat_1 tel que pour toute pile $s \in \mathrm{Stacks}_2(\Gamma)$ avec $d_s = d$ et pour toute pile $r \in \mathrm{Stacks}_1$, $d_{s\cdot r} = d'$ si et seulement si $\mathrm{top}_1(s) \in T_{d,d'}$. Il suffit de prendre $T_{d,d'}$ égal à l'union des intersections $\cap_{\ell\in[1,n]} R_\ell$ tel que pour tout $\ell \in [1,n]$, $\bullet \xrightarrow[A_\ell]{\mathrm{N}_1\cdot R_\ell} d'_\ell$ si $d = (\bullet,\ldots,\bullet)$ ou égal à l'union des intersections $\cap_{\ell\in[1,n]} R_\ell$ telles que pour tout $\ell \in [1,n]$, $d_\ell \xrightarrow[A_\ell]{\mathrm{N}_1\cdot R_\ell} d'_\ell$ sinon.

Nous construisons maintenant pour tout $d_0 \in D$, un automate fini $B_{d_0} = (Q_{d_0},I_{d_0},F_{d_0},\Delta_{d_0})$ étiqueté par $\Gamma_1 \cup \{\, 1 \,\} \cup \mathcal{T}_{\mathrm{Rat}_1(\Gamma)}$. Nous prenons $Q_d = Q_A \times Q_D$, $I_B = I_A \times \{\, d \,\}$ et $F_d = F_A \times D$. L'ensemble Δ_d des transitions est donné par:

$$\begin{aligned} \Delta_d \;=\; & \{(p,d) \xrightarrow{\gamma} (q,d) \mid d \in D,\, p \xrightarrow{\gamma} q \in A \text{ et } \gamma \in \Gamma_1\} \\ \cup\; & \{(p,d) \xrightarrow{T_L^2} (q,d) \mid d \in D,\, p \xrightarrow{T_{L_\ell}^2} q \in A, \ell \in [1,n] \text{ et } L \in T_d^\ell\} \\ \cup\; & \{(p,d) \xrightarrow{T_L^2\cdot 1} (q,d') \mid d,d' \in D,\, p \xrightarrow{1} q \in A \text{ et } L \in T_{d,d'}\}. \end{aligned}$$

Nous noterons R_{d_0} l'ensemble de $\mathrm{Rat}((\Gamma_1 \cup \{\, 1 \,\} \cup \mathcal{T}_{\mathrm{Rat}_1(\Gamma)})^*)$ accepté par B_{d_0}. On vérifie aisément que pour toute pile $s \in \mathrm{Stacks}_2(\Gamma)$ avec $d_s = d_0$, $\mathcal{R}(R)(s) = \mathcal{R}(R_{d_0})(s)$. Il suit alors que le langage R accepté par A est égal à: $\bigcup_{d\in D} T_{S_d}^2 \cdot R_d$.

Pour conclure, il suffit de remarquer par la proposition 4.5.4 que pour tout $R \in \mathrm{R}_1$, $\mathcal{D}^{\mathrm{c}}(R) = \mathcal{D}(R)$ est une union finie de relations dans $\mathcal{D}(\mathrm{Rew}_1) = \mathcal{D}^{\mathrm{c}}(\mathrm{Rew}_1)$. De plus, pour tout $R \in \mathrm{R}_1$, $\mathcal{R}^{\mathrm{c}}(R\cdot 1) = \mathcal{R}^{\mathrm{c}}(T^2_{\mathrm{Dom}(\mathcal{R}(R))}\cdot 1)$ et par la proposition 4.5.13, $\mathrm{Dom}(\mathcal{R}(R)) \in \mathrm{Rat}_1(\Gamma)$. □

En combinant les deux propositions précédentes, nous obtenons une représentation normalisée des ensembles de $\mathrm{CRat}_3(\Gamma)$.

Proposition 4.6.11. *Tout ensemble de R de $CRat_3(\Gamma)$ est égal à une union finie d'ensembles dans*

$$\mathcal{R}^{\mathrm{c}}\left(Rat\left((2\cdot Rew_2^{\mathrm{c}})^*\right)\right)\left(Rat_2(\Gamma)\right).$$

4.6.3 Inclusion stricte de $\mathrm{CRat}_3(\Gamma)$ dans $\mathrm{Rat}_3(\Gamma)$.

Dans ce sous-paragraphe, nous établissons que le langage S présenté au début du paragraphe n'appartient pas à CRat_3. Ce résultat, comme nous l'avons déjà remarqué, implique que CRat_3 est strictement inclus dans Rat_3 et que CRat_3 n'est pas une algèbre de Boole.

Proposition 4.6.12. *Le langage sur l'alphabet $\Gamma = \{a\}$,*

$$S = \{[[[a^{p_1}]\ldots[a^{p_m}]]_2[[a^{p_1}]\ldots[a^{p_n}]]_2]_3 \mid n \leq m \text{ et } p_n = p_{n-1}\},$$

n'appartient pas à l'ensemble $CRat_3(\Gamma)$.

Démonstration. Supposons par l'absurde que S appartienne à $\mathrm{CRat}_3(\Gamma)$. Par la proposition 4.6.11, il existe $R_1,\ldots,R_n \in \mathrm{Rat}_2(\Gamma)$, $U_1,\ldots,U_n \in \mathrm{Rat}((\mathcal{T}_{\mathrm{Rat}_1(\Gamma)}\cdot\overline{1})^*)$, $V_1,\ldots,V_n \in \mathrm{Rew}_1\cdot\mathrm{Rat}((1\cdot\mathrm{Rew}_1)^*)$ et $W_1,\ldots,W_n \in \mathrm{WRat}_2(\Gamma)$ tels que:

$$S = \bigcup_{\ell\in[1,n]} \mathcal{R}^{\mathrm{c}}\left(2\cdot U_\ell\cdot T^2_{W_\ell}\cdot V_\ell\right)(R_\ell).$$

Pour tout $\ell \in [1,n]$, nous noterons A_ℓ un automate normalisé faible acceptant W_ℓ (cf. proposition 4.6.11).

Avant de pouvoir aboutir à une contradiction, il nous faut établir un certain nombre de résultats préliminaires.

Fait 1. Pour tout $R \in \mathrm{Rew}_1$, il existe une constante k_R telle que pour toute pile $s \in \mathrm{Stacks}_2(\Gamma)$, s'il existe une pile $s' \in \mathrm{Stacks}_2(\Gamma)$ telle que $|s'| - |s| \geq k_R$ alors $(s,s') \in \mathcal{D}^{\mathrm{c}}(R)$ alors $\mathcal{R}(s)$ est infini.

Par définition de Rew_1, il existe R_1,R_2 et $R_3 \in \mathrm{Rat}_1(\Gamma)$ telle que R soit égale à $\overline{R_1}\cdot T_{R_2}\cdot R_3$. Il suffit de prendre k_R strictement supérieur au nombre minimal d'états d'un automate acceptant R_3. Supposons alors qu'il existe $s' \in \mathrm{Stacks}_2(\Gamma)$ telle que $|s'| - |s| \geq k_R$ et $(s,s') \in \mathcal{D}^{\mathrm{c}}(R)$. Il existe dans s_1,s_2 et s_3 appartenant respectivement à R_1,R_2 et R_3 tels que $s = s_2s_1$ et $s' = s_2s_3$. Nous avons $|s_3| \geq |s_3| - |s_1| = |s'| - |s| \geq k_R$ et par définition k_R, il suit que R_3 est infini et donc que $R(s)$ est infini.

Fait 2. Pour toute relation $V \in \mathrm{Rew}_1\cdot\mathrm{Rat}((1\cdot\mathrm{Rew}_1)^*)$, il existe une constante $k_V \in \mathbf{N}$ telle que pour toute pile $s = \left[s_1,\ldots,s_{|s|}\right]_2 \in \mathrm{Stacks}_2(\Gamma)$, s'il existe une

pile $t \in \left[t_1,\ldots,t_{|t|}\right]_2$ avec $|t_1| - |s_{|t|}| > k_0$ et $s \cdot t \in \mathcal{R}^{\mathrm{c}}(V)(s)$ alors $\mathcal{R}^{\mathrm{c}}(V)(s)$ est infini.

Soit $V \in R_0 \cdot P$ où $R_0 \in \mathrm{Rew}_1$ et $P \in \mathrm{Rat}((1 \cdot \mathrm{Rew}_1)^*)$. L'ensemble P est accepté par un automate $B = (Q_B,I_B,F_B,\Delta_B)$ étiqueté par $1 \cdot \mathrm{Rew}_1$.

Nous définissons pour tout $q \in Q_B$, l'automate $B_q = (Q_B,\{q\},F_B,\Delta_B)$. La relation $\mathcal{R}^c(\mathcal{L}(B_q))$ induite par B a pour domaine un ensemble rationnel noté D_q.

Pour tout $R \in Rew_1$ et pour tout état $q \in Q_B$, la relation $\mathcal{D}^{\mathrm{c}}(R)$ restreinte en image à D_q est égale à l'union finie $\bigcup_{i\in I} \mathcal{D}(R_i)$ où pour tout $i \in I$, $R_i \in \mathrm{Rew}_1$ (cf. proposition 4.5.4). Nous noterons $k_{R,q}$ le maximum de l'ensemble $\{k_{R_i} \mid i \in I\}$.

Nous prenons k_V supérieure au maximum sur tout $R \in \mathrm{Rew}_1$ apparaissant dans une étiquette de B et tout $q \in Q_B$ des constantes $k_{R,q}$.

Soit s une pile de $\mathrm{Stacks}_2(\Gamma)$. Supposons qu'il existe une pile $t \in \left[t_1,\ldots,t_{|t|}\right]_2$ avec $|t_1| - |s_{|t|}| > k_V$ et telle que $s \cdot t \in \mathcal{R}^{\mathrm{c}}(V)(s)$. Il existe donc $i_0 \in I_B$, $q \in Q_B$ et $i_0 \xrightarrow{1\cdot R} Bq \in \Delta_B$ telles que $s \cdot t_1 \in (\mathcal{D}^{\mathrm{c}}(1 \cdot R))(s)$ et $s \cdot t \in (\mathcal{D}^{\mathrm{c}}(\mathcal{L}(B_q)))(s \cdot t_1)$. En particulier $(s_{|s|},t_1)$ appartient à la relation R restreinte en image à D_q (notée $R|_{D_q}$). Par définition, la constante k_v est supérieure à $k_{R,q}$. Il suit donc par le fait 1 que $R|_{D_q}(s_{|s|})$ est donc infini. Il suit donc que $\mathcal{R}^{\mathrm{c}}(V)$ est infini.

Nous dérivons maintenant la contradiction.

Considérons l'ensemble T de tous les ensembles rationnels de $\mathrm{Rat}_1(\Gamma)$ apparaissant dans les instructions de tests des langages $U_1,\ldots,U_n$, ou dans les étiquettes des automates $A_1,\ldots,A_n$ ou égaux à l'un des ensembles de la suite $\mathrm{top}_1(\mathrm{Dom}(\mathcal{R}(V_1))),\ldots,\mathrm{top}_1(\mathrm{Dom}(\mathcal{R}(V_n)))$. Par définition, l'ensemble T est fini et il existe donc deux piles u_0 et u_1 telles que:

- $|u_1| - |u_0| \geq k_0$ où k_0 est le maximum de k_{V_i} pour $i \in [1,n]$,
- $\{R \in T \mid u_0 \in R\} = \{R \in T \mid u_1 \in T\}$.

Intuitivement, les deux piles u_0 et u_1 sont indistinguables pour les U_i et les W_i. Nous considérons uniquement des piles de niveau 2 composées de piles de niveau 1 égales à u_0 ou bien u_1. Nous noterons X l'ensemble de ces piles (*i.e.* $X = \{s = [s_1,\ldots,s_n]_2 \mid \text{pour tout } \ell \in [1,n], s_\ell \in \{u_0,u_1\}\}$).

Par définition de T et des piles u_0 et u_1, il suit que, pour toutes piles s et s' dans X avec $|s| = |s'|$ et pour tout $\ell \in [1,n]$, $s \in W_\ell$ si et seulement si $s' \in W_\ell$. De même, pour toutes piles $s = [s_1,\ldots,s_m]_2$ et $s' = [s'_1,\ldots,s'_m]_2$ dans X, pour tout $p \leq m$ et pour tout $\ell \in [1,n]$, $[s_1,\ldots,s_p]_2 = \mathcal{R}^{\mathrm{c}}(U_\ell)(s)$ si et seulement si $\left[s'_1,\ldots,s'_p\right]_2 = \mathcal{R}^{\mathrm{c}}(U_\ell)(s')$.

Fait 3. Il existe trois piles $s = \left[s_1,\ldots,s_{|s|}\right]_2, r = \left[r_1,\ldots,r_{|r|}\right]_2, t = \left[t_1,\ldots,t_{|t|}\right]_2$ appartenant à X avec $|r| = |t| \geq 2$, $s_{|s|} = r_1 = r_2 = t_2 = u_1$ et $t_1 = u_0$ pour tout $\ell \in [1,n]$, $s \cdot t \in R_\ell$ et $s \cdot r \in R_\ell$.

À chaque pile $s \in X$, nous associons l'ensemble $\Psi(s) = \{\ell \in [1,n] \mid s \in R_\ell\}$. Soit $m = 2(2^n + 1)$. Pour tout $\ell \in [1,2^n + 1]$, nous définissons la pile $s^\ell = \left[s^\ell_1,\ldots,s^\ell_m\right] \in X$ en prenant pour tout $i \neq 2\ell \in [1,m]$, $s_i = u_1$ et $s_{2\ell} = u_0$. Il

existe donc $i < j \in [1,2^n]$ tel que $\Psi(s_i) = \Psi(s_j)$. Il existe donc trois piles s,t et $r \in X$ telles que $s_i = s \cdot t$ et $s_j = s \cdot r$ et satisfaisant les conditions de l'énoncé.

Par définition, la pile $[\, s \cdot r, s \cdot r_1 \,]_3$ appartient à S. Il existe donc $\ell_0 \in [1,n]$ et une pile $s' \in X$ tels que $s \cdot r \in R_{\ell_0}$, et que $(s \cdot r, s \cdot r_1) \in \mathcal{R}^{\mathrm{c}}(U_{\ell_0} \cdot T_{W_{\ell_0}} \cdot V_{\ell_0})(s \cdot r)$.

Il existe donc une pile $s' \in X$ appartenant à $\mathcal{R}^{\mathrm{c}}(U_{\ell_0})(s \cdot r) \cap T_{W_{\ell_0}}$ et telle que $(s', s \cdot r_1)$ appartient à $\mathcal{R}^{\mathrm{c}}(V_{\ell_0})$. Nous distinguons deux cas selon la valeur de s'.

Cas $s' = s \cdot r_1$**.**

La pile $s \cdot t$ appartient aussi à R_{ℓ_0} et $t' = s \cdot t_1$ appartient à $\mathcal{R}^{\mathrm{c}}(U_{\ell_0})(s \cdot t) \cap T_{W_{\ell_0}}$. Comme $r_1 = u_1$ appartient à $\mathrm{top}_1(\mathrm{Dom}(\mathcal{R}^{\mathrm{c}}(V_{\ell_0})))$, il suit que u_0 appartient aussi à $\mathrm{top}_1(\mathrm{Dom}(\mathcal{R}^{\mathrm{c}}(V_{\ell_0})))$. Il existe donc une pile $t' \in X$ telle que $(s \cdot t_1, s \cdot t') \in \mathcal{R}^{\mathrm{c}}(V_{\ell_0})$. Donc $(s \cdot t, s \cdot t')$ appartient à S et par définition de S, $t_1 = t'_1 = u_0$ et $t'_2 = u_1$. Par le fait 2 et par définition de k_0, il suit que $\mathcal{R}^{\mathrm{c}}(V_{\ell_0})(s_0)$ est infini. Ce qui contredit la définition de S.

Cas $s' = [\, s_1, \ldots, s_p \,]_2$ **pour un certain** $p \leq |s|$**.**

La pile $s \cdot t$ appartient aussi à R_{ℓ_0} et s' appartient à $\mathcal{R}^{\mathrm{c}}(U_{\ell_0})(s \cdot t) \cap T_{W_{\ell_0}}$. Il suit donc que $[\, s \cdot t, s \cdot r_1 \,]_3$ que appartient S; ce qui amène la contradiction. □

4.7 Caractérisation par définissabilité logique

Dans ce paragraphe, nous considérons un graphe «canonique» associé aux piles de niveau k avec le jeux d'opérations Ops_k. Ce graphe, noté $\mathrm{GStacks}_k$, a pour sommets les piles de niveau k sur Γ et est étiqueté par l'ensemble d'instructions $\Sigma_k = \Gamma \cup [1,k-1]$. Pour clarifier notre présentation, nous fixons un alphabet de pile Γ égal à $\{\, a,b \,\}$.

$$\mathrm{GStacks}_k = \{(s,i,\mathcal{R}(i)(s) \mid s \in \mathrm{Stacks}_k(\Gamma) \text{ et } i \in \Sigma_k\}.$$

Au niveau 1, $\mathrm{GStacks}_1$ est isomorphe à l'arbre binaire complet étiqueté par Γ. La figure 4.10 présente le graphe $\mathrm{GStacks}_2$. Il est aisé de vérifier que pour tout $k \geq 1$, $\mathrm{GStacks}_{k+1} = \mathrm{Treegraph}(\mathrm{GStacks}_k,k)$. De plus, pour toute opération $\rho \in \mathrm{Ops}_k$, il existe une formule monadique $\varphi_\rho(x,y)$ telle que pour toutes piles s et $s' \in \mathrm{Stacks}_k$, $\mathrm{GStacks}_k \models \varphi_\rho[s,s']$ si et seulement si $\rho(s)$ est définie et égale à s'. Il est alors immédiat que les ensembles rationnels de piles de niveau k sont définissables en logique monadique sur $\mathrm{GStacks}_k$ (*i.e.* pour tout $R \in \mathrm{Rat}_k$, il existe une formule monadique φ_R telle que R soit égal à l'ensemble $\{s \in \mathrm{Stacks}_k \mid \mathrm{GStacks}_k \models \varphi_R(s)\}$.

Dans ce paragraphe, nous établissons la réciproque : les ensembles définissable en logique monadique sur $\mathrm{GStacks}_k$ sont des ensembles rationnels de piles de niveau k. Au niveau 1, cette propriété découle de la caractérisation de la logique monadique par des automates d'arbres à parité (cf. paragraphe 1.4.4). Pour étendre

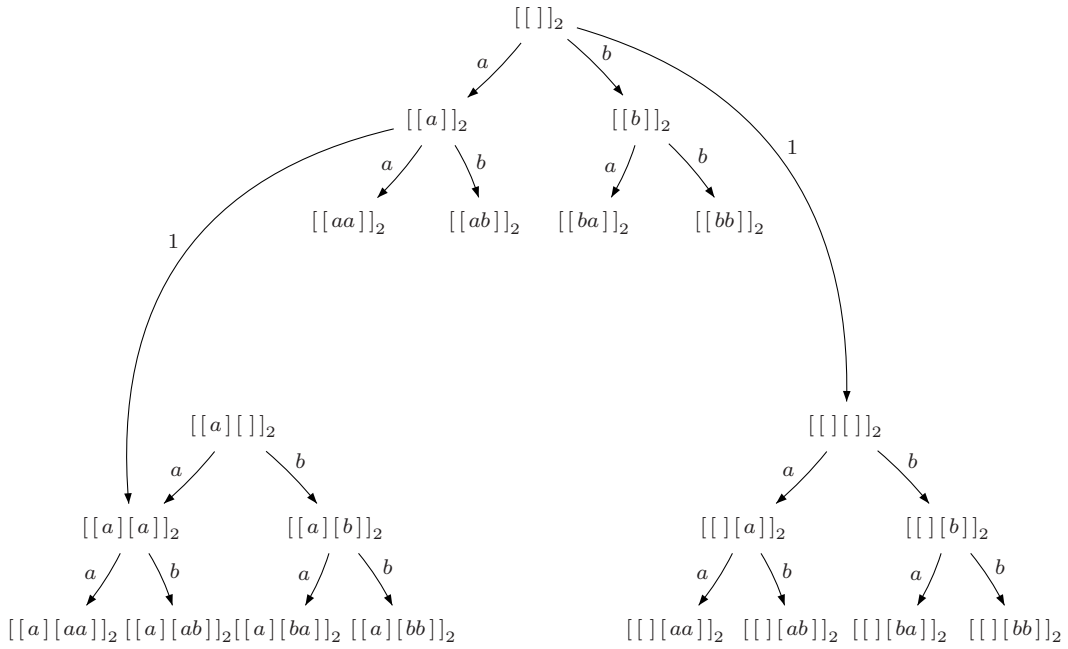


FIG. 4.10 – *Le graphe* $\mathrm{GStacks}_2$.

cette propriété à tout niveau $k \geq 2$, il est nécessaire de considérer un arbre «canonique» associé aux piles de niveau k. Cet arbre noté $\mathrm{TStacks}_k$ et ses propriétés sont présentées dans le sous-paragraphe 4.7.1. Le sous-paragraphe 4.7.2 établit que les ensembles MSO-définassables sur le graphe $\mathrm{GStacks}_k$ sont les ensembles de Rat_k. À l'aide de ce résultat, nous établissons que les relations (binaires) MSO-définissables sur $\mathrm{GStacks}_k$ sont les relations préfixes-reconnaissables de niveau k (cf. paragraphe 4.5). Enfin dans le sous-paragraphe 4.7.3, nous étendons le lemme des bases de Rabin (cf. théorème 1.4.10) à tous les graphes $\mathrm{GStacks}_k$.

4.7.1 L' arbre $\mathrm{TStacks}_k$

L'arbre le plus naturel associé aux piles de niveau k, noté $\mathrm{TStacks}_k$, est l'arbre déterministe étiqueté par Γ_k^{o} ayant pour racine la pile vide de niveau k et tel que pour toute pile $s \in \mathrm{Stacks}_k$ l'étiquette du chemin partant de la pile vide à s soit la suite réduite de s. Formellement, l'arbre $\mathrm{TStacks}_k$ est défini par:

$$\mathrm{TStacks}_k = \{(s,\gamma,s' \mid s,s' \in \mathrm{Stacks}_k, \gamma \in \Gamma_k^{\mathrm{o}} \text{ et } \rho_{s'} = \rho_s\gamma\}.$$

Par unicité de la suite réduite (cf. proposition 4.1.9), $\mathrm{TStacks}_k$ est un arbre déterministe de racine $[\,]_k$.

Pour tout niveau $k \geq 1$, l'arbre $\mathrm{TStacks}_k$ peut s'interpréter dans le graphe $\mathrm{GStacks}_k$ et vice et versa.

Proposition 4.7.1. *Pour tout niveau $k \geq 1$, il existe deux interprétations monadiques $\mathcal{I}_k$ et $\mathcal{J}_k$ telle que* $\mathrm{GStacks}_k = \mathcal{I}_k(\mathrm{TStacks}_k)$ *et* $\mathrm{TStacks}_k = \mathcal{J}_k(\mathrm{GStacks}_k)$.

Démonstration. Soit $k \geq 1$. Considérons l'interprétation $\mathcal{I}_k = (\varphi_\gamma(x,y))_{\gamma \in \Sigma_k}$ où pour tout $\gamma \in \Sigma_k$, $\varphi_\gamma(x,y) = x \xrightarrow{\gamma} y \wedge y \xrightarrow{\overline{\gamma}} x$. Il est aisé de vérifier que $\mathrm{GStacks}_k = \mathcal{I}_k(\mathrm{TStacks}_k)$.
Considérons l'interprétation $\mathcal{J}_k = (\varphi_\gamma(x,y))_{\gamma \in \Gamma_k^o}$ où pour tout $\gamma \in \Gamma_k^o$,

$$\begin{aligned} \varphi_\gamma(x,y) &= \exists r, \psi(r) \wedge \bigvee_{\gamma' \neq \overline{\gamma} \in \Gamma_k^o} r \overset{\mathrm{Red}_k^{\gamma'}}{\Longrightarrow} x \wedge x \xrightarrow{\gamma} y \\ \varphi_{\overline{\gamma}}(x,y) &= \exists r, \psi(r) \wedge \bigvee_{\gamma' \neq \gamma \in \Gamma_k^o} r \overset{\mathrm{Red}_k^{\gamma'}}{\Longrightarrow} x \wedge y \xrightarrow{\gamma} x \end{aligned}$$

avec $\gamma \in \Sigma_k$, $\psi(x)$ une formule monadique définissant la pile vide de niveau k dans $\mathrm{GStacks}_k$ et où $\mathrm{Red}_k^{\gamma'}$ désigne l'ensemble rationnel Red_k réduit aux suites se terminant par l'instruction γ' ou vides. On vérifie que $\mathrm{TStacks}_k = \mathcal{J}_k(\mathrm{GStacks}_k)$. □

Une conséquence immédiate de cette proposition et de la proposition 3.1.3 est que les ensembles définissables en logique monadique dans $\mathrm{GStacks}_k$ et dans $\mathrm{TStacks}_k$ coïncidents. Il en va de même pour les relations définissables dans ces deux graphes.

Nous présentons maintenant une série de transformations de graphes permettant d'obtenir $\mathrm{TStacks}_{k+1}$ à partir de $\mathrm{TStacks}_k$. Cette construction est une simple adaptation du lemme 3.3.2.

Lemme 4.7.2. *Il existe une substitution finie inverse h^{-1} et une restriction monadique $\mathcal{R}$ telles que*

$$\mathrm{TStacks}_{k+1} \approx \mathcal{R}(\mathrm{Unf}(h^{-1}(\mathrm{TStacks}_k), [\ ]_k)).$$

Démonstration. Considérons la substitution h^{-1} définie pour tout $c \in \Gamma_k^o \cup \{ k \}$, par:

$$\begin{cases} h_1(\gamma) = \{ \gamma \} & \gamma \in \Gamma_k^o \\ h_1(k) = \{ \varepsilon \} \end{cases}$$

Pour tout arc (s,γ,s') étiqueté par $\gamma \in \Gamma_k^o$, l'application de h^{-1} conserve cet arc et rajoute l'arc inverse $(s',\overline{\gamma},s)$ étiqueté par $\overline{\gamma}$.

La restriction monadique $\mathcal{R}$ ne conserve que les sommets accessibles depuis la racine par un chemin dont l'étiquette ne contient pas de facteurs de la forme $\gamma\overline{\gamma}$ pour $\gamma \in \Gamma_k^o$.

Il est aisé de vérifier que $\mathrm{TStacks}_{k+1}$ est isomorphe à $\mathcal{R}(\mathrm{Unf}(h^{-1}(\mathrm{TStacks}_k)))$. □

4.7.2 MSO-définissabilité sur $\mathrm{GStacks}_k$.

La proposition 4.7.2 permet d'utiliser les résultats du chapitre 3 pour définir les ensembles définissables en logique monadique sur $\mathrm{TStacks}_{k+1}$ en fonction des ensembles définissables en logique monadique sur $\mathrm{TStacks}_k$.

Proposition 4.7.3. *Les ensembles définissables en logique monadique sur* $\mathrm{TStacks}_k$ *sont des ensembles rationnels de* Rat_k.

Démonstration. La preuve procède par induction sur le niveau $k \geq 1$. Au niveau 1, ce résultat se découle du théorème 1.4.10. Supposons le résultat vrai au niveau $k \geq 1$. Montrons qu'il est vrai au niveau $k+1$.

Par le lemme 4.7.2, il existe une substitution finie inverse h^{-1} et une restriction monadique $\mathcal{D} = (\delta(x))$ telles que:

$$\mathrm{TStacks}_{k+1} \approx \mathcal{D}(\mathrm{Unf}(h^{-1}(\mathrm{TStacks}_k), [\ ]_k)).$$

Soit $\varphi(x)$ une formule monadique. Considérons le marquage monadique $\mathcal{M}$ associant la couleur \$ aux sommets satisfaisant la formule $\varphi(x) \wedge \delta(x)$. Par la proposition 3.4.4, il existe un coloriage monadique $\mathcal{M}' = (\varphi_c(x))_{c \in C}$ pour un certain sous-ensemble $C \subseteq \Theta$ et un coloriage rationnel μ défini par $\mu(\$) = R$ pour $R \in \mathrm{Rat}(\Gamma_k^o \cup \{\, k \,\} \cup C)^*$ tels que:

$$\mathcal{M}(\mathrm{Unf}(h^{-1}(\mathrm{TStacks}_k))) = \mu_{[\ ]_k}(\mathrm{Unf}(\mathcal{M}'(h^{-1}(\mathrm{TStacks}_k)))).$$

Il existe un coloriage monadique $\mathcal{M}'' = (\psi_c(x))_{c \in C}$ tel que $\mathcal{M}'(h^{-1}(\mathrm{TStacks}_k)) = h^{-1}(\mathcal{M}''(\mathrm{TStacks}_k))$. Par hypothèse de récurrence, pour tout $c \in C$, l'ensemble $R_c = \{s \in \mathrm{Stacks}_k \mid \mathrm{TStacks}_k \models \psi_c[s]\}$ est un ensemble rationnel de piles de niveau k. Considérons le langage R' dans $\mathrm{Rat}(\Gamma_k^o \cup \{\, k \,\} \cup \{T_{R_c} \mid c \in C\})$ obtenu en remplacant dans R les occurences de c par T_{R_c} pour tout $c \in C$. Il est aisé de vérifier que l'ensemble $S = \{s \in \mathrm{Stacks}_{k+1} \mid \mathrm{TStacks}_{k+1} \models \varphi[s]\}$ est égal à $\mathcal{R}(R')([\ ]_{k+1})$. □

Par la proposition 4.7.1, il suit que les ensembles définissables dans $\mathrm{GStacks}_k$ sont des ensembles rationnels de piles de niveau k. Comme nous l'avons déjà mentionné, les ensembles rationnels de Rat_k sont définissables dans $\mathrm{GStacks}_k$.

Théorème 4.7.4. *Les ensembles définissables en logique monadique sur* $\mathrm{GStacks}_k$ *sont les ensembles de* Rat_k.

Nous pouvons grâce à ce résultat et à la proposition 3.2.1 donner une caractérisation des relations définissables dans $\mathrm{GStacks}_k$. Pour le niveau 1, cette caractérisation a été établie dans [Bar97].

Théorème 4.7.5. *Les relations définissables en logique monadique sur* $\mathrm{GStacks}_k$ *sont les relations préfixe-reconnaissables de niveau* k.

Démonstration. Par définition de PR_k, pour toute relation $R \in \mathrm{PR}_k$, il il existe une formule monadique $\varphi_R(x,y)$ telle que pour toute piles $s,s' \in \mathrm{Stacks}_k$, $(s,s') \in R$ si et seulement si $\mathrm{GStacks} \models \varphi_R[s,s']$.

Pour la réciproque, il suffit d'établir que les relations définissables dans $\mathrm{TStacks}_k$ appartiennent à PR_k. Soit $\varphi(x,y)$ une formule monadique. Par la proposition 3.2.1, il existe un coloriage monadique $\mathcal{M}' = (\varphi_c(x))_{c\in C}$ pour un ensemble fini $C \subset \Theta$ et un ensemble R dans $\mathrm{Rat}(\Gamma_k^{\mathrm{o}} \cup C)^*$ tels que pour toutes piles $s,s' \in \mathrm{Stacks}_k$,

$$\mathrm{TStacks}_k \models \varphi[s,s'] \Leftrightarrow s \xRightarrow{R} s'.$$

Par le théorème 4.7.4, pour tout $c \in C$ l'ensemble $R_c = \{s \in \mathrm{Stacks}_k \mid \mathrm{TStacks}_k \models \varphi_c[s]\}$ appartient à Rat_k. Considérons le langage R' dans $\mathrm{Rat}(\Gamma_k^{\mathrm{o}} \cup \{T_{R_c} \mid c \in C\})$ obtenu en remplaçant dans R les occurences de c par T_{R_c} pour tout $c \in C$. Nous avons pour toutes piles $s,s' \in \mathrm{Stacks}_k$,

$$s \xRightarrow{R} s' \;\Leftrightarrow\; s' \in \mathcal{R}(R')(s).$$

La relation définie par la formule $\varphi(x,y)$ sur TStacks appartient à PR_k. □

4.7.3 Sélection sur $\mathrm{GStacks}_k$.

Dans ce sous-paragraphe, nous établissons que les graphes $\mathrm{GStacks}_k$ possèdent la propriété de sélection. Ce résultat a été obtenu par Fratani dans [Fra05].

Théorème 4.7.6 ([Fra05])**.** *Pour tout $k \geq 1$, le graphe $\mathrm{GStacks}_k$ possède la propriété de sélection.*

Démonstration. La preuve procède par récurrence sur le niveau $k \geq 1$. Comme nous l'avons mentionné dans le paragraphe 1.4.6, l'arbre binaire complet possède la propriété de sélection; ce qui établit le cas de base. Supposons que la propriété est vrai au niveau k et montrons quelle est vraie au niveau $k+1$.

Par la proposition 3.5.2 et la proposition 4.7.1, nous avons que $\mathrm{GStacks}_k$ possède la propriété de sélection si et seulement si $\mathrm{TStacks}_k$ la possède. Par hypothèse de récurrence, $\mathrm{TStacks}_k$ satisfait la propriété de sélection. Par la proposition 3.5.3, $\mathrm{Treegraph}(\mathrm{TStacks}_k,k)$ possède la propriété de sélection. Par la proposition 3.4.1 et la proposition 4.7.1, il existe deux interprétations monadiques $\mathcal{I}$ et $\mathcal{J}$ telles que $\mathrm{Treegraph}(\mathrm{TStacks}_k,k) \approx \mathcal{I}(\mathrm{Treegraph}(\mathrm{GStacks}_k,k))$ et $\mathcal{J}(\mathrm{Treegraph}(\mathrm{TStacks}_k,k)) \approx \mathrm{Treegraph}(\mathrm{GStacks}_k,k))$. Rappelons que que nous avons l'égalité $\mathrm{Treegraph}(\mathrm{GStacks}_k,k) = \mathrm{GStacks}_{k+1}$. Par la proposition 3.5.2, $\mathrm{GStacks}_{k+1}$ possède la propriété de sélection. □

En combinant les théorèmes 4.7.4 et 4.7.6, nous obtenons l'extension suivante du théorème 1.4.10 aux graphes $\mathrm{GStacks}_k$.

Théorème 4.7.7. *Pour tout $k \geq 1$ et pour toute formule monadique $\varphi(X)$ telle que $\mathrm{GStacks}_k \models \exists X, \varphi(X)$, il existe un ensemble $R_\varphi \in Rat_k$ tel que $\mathrm{GStacks}_k \models \varphi[R_\varphi]$.*

4.8 Lien avec les automates sur les mots

Nous concluons en montrant comment les résultats de normalisation obtenus dans ce chapitre peuvent être utilisés pour étudier des extensions des automates de mots. Nous considérons des enrichissements des automates finis sur les mots qui ont la même expressivité mais ont une plus grande concision en terme du nombre d'états. Nous présentons des enrichissements classiques tels que la bidirectionalité [HU79], l'alternance [CKS81] et l'ajout de galets [BH67, GGK91, GH96, EH99]. Pour une présentation détaillée des résultats sur ces modèles d'automates et des complexités des transformations passant des uns aux autres, nous renvoyons le lecteur à [GH96]. Nous considérons différentes combinaisons de ces enrichissements et nous montrons comment elles peuvent être simulées par des automates sur Γ_k.

Avant de passer à la présentation des différents modèles d'automates, il nous faut préciser ce que nous entendons par «simulation». En effet, un automate sur Γ_k acceptant des langages de piles de niveaux k qui sont des mots de mots ... Il existe cependant un codage naturel, noté Ψ_k, d'un mot sur Γ en une pile de $\mathrm{Stacks}_k(\Gamma)$. Cet encodage est l'identité au niveau 1 et au niveau $k+1 \geq 2$, $\Psi_k(w) = [\,\Psi_{k-1}(w)\,]_k$ pour tout $w \in \Gamma^*$.
Pour simuler au niveau k des automates sur les mots, nous ne considérerons dans les langages acceptés par les automates sur Γ_k que les piles correspondant au codage par Ψ_k d'un mot sur Γ^*. Ainsi, pour tout automate A sur Γ_k, nous noterons $\mathcal{M}(A)$ le langage de mots sur Γ^* égal à $\Psi_k^{-1}(\mathcal{S}(A))$. Dans la suite, nous ne considérerons que des automates sur Γ_k tels que $\mathcal{S}(A) = \Psi_k(\mathcal{M}(A))$. Nous étendons cette notation aux automates alternants sur Γ_k et aux automates avec tests. Des exemples de tels automates au niveau 2 sont donnés dans l'exemple 4.3.8

Proposition 4.8.1. *Pour tout automate A sur Γ_k, tout automate B alternant sur Γ_k et tout automate C sur Γ_k avec tests dans $Rat_k(\Gamma)$, les langages de mots $\mathcal{M}(A)$, $\mathcal{M}(B)$ et $\mathcal{M}(C)$ sont rationnels.*

Démonstration. Nous avons vu dans la proposition 4.3.5, le théorème 4.3.43 et dans le corollaire 4.3.44, que les langages acceptés par ces différents modèles d'automates appartiennent tous à $\mathrm{Rat}_k(\Gamma)$. Pour tout automate A de niveau k, $\mathcal{M}(A) = \mathrm{top}_1(\mathcal{S}(A) \cap \Psi_k(\Gamma^*))$. Comme $\Psi_k(\Gamma^*)$ est égal à $\mathcal{R}_k(\Gamma^*)([\,]_k)$, il suit que

$\Psi_k(\Gamma^*)$ appartient à $\mathrm{Rat}_k(\Gamma)$. Donc comme $\mathrm{Rat}_k(\Gamma)$ est fermé par intersection, il suit par le corollaire 4.3.51 que $\mathcal{M}(A)$ est un langage rationnel de mots sur Γ^*. □

En utilisant les résultats des paragraphes précédents, nous pouvons préciser la complexité de la transformation d'un automate A de niveau k en un automate fini sur Γ acceptant $\mathcal{M}(A)$.

Théorème 4.8.2. *Pour tout niveau $k \geq 2$ et pour tout alphabet fini Γ, nous avons:*

1. *Pour tout automate A de mots sur Γ_k, il existe un automate B fini sur Γ acceptant le même langage. De plus $|B|$ est borné par* $\exp[k-1](|A|)$.
2. *Pour tout automate A de mots sur Γ_k avec tests dans un ensemble fini $\mathcal{L} \subset Rat_k(\Gamma)$, il existe un automate B fini sur Γ acceptant le même langage. De plus, si pour tout $L \in \mathcal{L}$, L est accepté par un automate entièrement déterministe et complet $|B|$ est borné par* $\exp[k-1](|A| + \prod_{L\in\mathcal{L}} |A_L|)$.
3. *Pour tout automate A de mots alternant sur Γ_k, il existe un automate fini déterministe sur Γ acceptant le même langage. De plus, $|B|$ est borné par* $\exp[k](|A|)$.

Démonstration. D'après les théorèmes 4.4.15 et 4.4.25, nous pouvons nous concentrer sur des automates entièrement déterministes et complets sur Γ_k pour $k \geq 2$.

Soit $A = (B,(B_L)_{L\in\mathcal{L}})$ une automate déterministe et complet sur Γ_k avec $k \geq 1$. Considérons l'automate C sur Γ avec tests dans $\{\mathcal{M}(B_L) \mid L \in \mathcal{L}\}$ obtenu en restreignant l'automate B aux transitions étiquetées par Γ et en remplaçant les instructions de tests $T_{\mathcal{S}(B_L)}$ par $T_{\mathcal{M}(B_L)}$. Il est aisé de vérifier que $\mathcal{M}(A)$ est égal à $\mathcal{S}(C)$. Supposons que pour tout $L \in \mathcal{L}$, $\mathcal{M}(B_L)$ est accepté par un automate C_L déterministe et complet sur Γ. Par la proposition 4.4.24, $\mathcal{M}(B)$ est accepté par un automate déterministe et complet sur Γ de taille bornée par $\exp[0](|B| + \prod_{L\in\mathcal{L}} |C_L|)$.

Nous allons traiter le cas des automates sur Γ_k. Les autres cas sont similaires. Soient $k \geq 2$ et A un automate sur Γ_k. Par le théorème 4.4.15, il existe un automate entièrement déterministe et complet $(B,(B_L)_{L\in\mathcal{L}})$ acceptant A et de taille bornée par $\exp[k-1](|A|)$. Par la remarque 4.4.16, la taille de tous les ensembles de tests intervenant dans cet automate est bornée par $\exp[k-2](|A|)$. Une récurrence immédiate utilisant le principe présenté au début de cette preuve établie l'existence d'un automate déterministe et complet sur Γ acceptant $\mathcal{M}(A)$ et de taille bornée par $\exp[k-1](|A|)$. □

4.8.1 Automates bidirectionnels

Nous introduisons, dans ce sous-paragraphe, les automates (non-déterministes) bidirectionnels [HU79] et nous montrons comment les simuler par des automates

sur Γ_2. Plus précisément, pour tout automate bidirectionnel A, nous construirons un automate B sur Γ_2 avec tests dans $\mathcal{L} \subset \mathrm{Rat}_2(\Gamma)$ de taille polynomiale en la taille A tel que $\mathcal{M}(B)$ est le langage de mots acceptés par A. L'ensemble de tests $\mathcal{L}$ ne dépend que de l'alphabet Γ et ne dépend pas de l'automate A. Par le théorème 4.8.2, nous déduirons que pour tout automate bidirectionnel A, il existe un automate déterministe et complet sur Γ acceptant $\mathcal{L}(A)$ de taille exponentielle en la taille de A. Ce résultat a été obtenu pour la première fois dans [She59].

Intuitivement, un automate bidirectionnel est une machine de Turing qui ne peut pas écrire sur sa bande. Nous allons présenter une définition des ces automates qui facilite la simulation par des automates sur Γ_2.

Définition 4.8.3. Un automate bidirectionnel sur Γ^* est un uplet (Q,I,F,Δ) où Q est un ensemble fini d'états, I et $F \subseteq Q$ sont respectivement l'ensemble des états initiaux et finaux et où Δ est un sous-ensemble de $Q \times \Gamma \times Q \times \mathcal{I}$ où $\mathcal{I}$ est l'ensemble des instructions $\{$ Last,First,¬Last,¬First,Right,Left $\}$.

Pour des raisons techniques, nous ne considérerons que des automates bidirectionnels ne travaillant pas sur une entrée vide. Une configuration est donc un uplet (w,i,q) où $w \in \Gamma^+$, $i \in [1,|w|]$ et où $q \in Q$. Intuitivement la configuration (w,i,q) représente une bande w avec la tête de lecture dans l'état q sur la $i^{\text{ème}}$ case. Les instructions de $\mathcal{I}$ sont des fonctions partielles sur l'ensemble $\{(w,i) \mid w \in \Gamma^+ \text{ et } i \in [1,|w|]\}$. L'instruction Right (resp. Left) déplace la tête de lecture d'une case vers la droite (resp. gauche) si la position i n'est pas égale à $|w|$ (resp. n'est pas égale à 1). Les instructions First,¬First,Last et $\neq$ Last ne modifient pas la configuration courante mais ne sont définies respectivement que si la position courante est égale à 1, différente de 1, égale à $|w|$ ou différente de $|w|$.

Une transition (p,γ,q,ρ) peut s'appliquer à une configuration (w,i,p) pour donner une configuration (w,j,q) si $w(i)$ est égal à γ et si l'instruction ρ appliquée à (w,i) est définie et égale à (w,j).

Nous dirons qu'un automate bidirectionnel A accepte un mot $w \in \Gamma^+$ s'il existe un calcul de A partant d'une configuration initiale $(w,1,q_i)$ avec $q_i \in I$ et terminant dans une configuration finale (w,j,q_f) pour $q_f \in F$. Nous noterons $\mathcal{L}(A)$ le langage accepté par A.

Nous présentons maintenant la simulation de ces automates par des automates sur Γ_2 avec tests dans $\mathrm{Rat}_2(\Gamma)$. Fixons un automate bidirectionnel $A = (Q,I,F,\Delta)$. Une configuration (w,i,q) de l'automate A sera «simulée» par la configuration $(q, [\,[\,w\,]\,[\,w(1),\ldots,w(i)\,]\,]_2)$ de l'automate B. Pour tout (w,i) avec $w \in \Gamma^+$ et $i \in [1,|w|]$, nous notons $[\![\,(w,i)\,]\!]$ la pile $[\,[\,w\,]\,[\,w(1),\ldots,w(i)\,]\,]_2$ de $\mathrm{Stacks}_2(\Gamma)$.

Avant de simuler les instructions de $\mathcal{I}$, nous introduisons des langages de tests dans $\mathrm{Rat}_2(\Gamma)$ qui sont $\mathrm{Conf} = \{[\,[\,w\,]\,[\,w'\,]\,]_2 \mid w,w' \in \Gamma^+ \text{ et } w' \sqsubseteq w\}$, $\mathrm{First} = \{[\,[\,w\,]\,[\,w(1)\,]\,]_2 \mid w \in \Gamma^+\}$, $\mathrm{Last} = \{[\,[\,w\,]\,[\,w\,]\,]_2 \mid w \in \Gamma^+\}$ ainsi que

les complémentaires de ces deux derniers langages notés respectivement First^{c} et Last^{c}. Nous vérifions aisément que ces langages sont acceptées par des automates entièrement déterministes et complets de taille polynomiale en la taille de l'alphabet Γ. Notons $\mathcal{L}_2$ l'ensemble de ces langages.

Nous associons à chaque instruction $\rho \in \mathcal{I}$, un ensemble fini non-ambigu $R_\rho \subset (\Gamma_2 \cup \mathcal{T}_{\mathcal{L}_2})^*$ simulant ρ. Nous adoptons les mêmes raccourcis de notations que dans le sous-paragraphe 4.1.5.2.

$$\begin{array}{lcllcl} R_{\text{Right}} & = & \{\gamma \cdot \mathcal{T}_{\text{Conf}} \mid \gamma \in \Gamma\} & R_{\text{Left}} & = & \{\bar{\gamma} \cdot \mathcal{T}_{\text{Conf}} \mid \gamma \in \Gamma\} \\ R_{\text{First}} & = & \{\, \mathcal{T}_{\text{First}} \,\} & R_{\neg\text{First}} & = & \{\, \mathcal{T}_{\text{First}^{\text{c}}} \,\} \\ R_{\text{Last}} & = & \{\, \mathcal{T}_{\text{Last}} \,\} & R_{\neg\text{First}} & = & \{\, \mathcal{T}_{\text{Last}^{\text{c}}} \,\}. \end{array}$$

Il est aisé de vérifier que pour tout $\rho \in \mathcal{I}$ et pour tout (w,i) avec $w \in \Gamma^+$ et $i \in [1,|w|]$, $\rho((w,i))$ est défini et égal à (w,j) si et seulement si $\mathcal{R}(R_\rho)(\llbracket\, (w,i) \,\rrbracket)$ est défini et égal à $\llbracket\, (w,j) \,\rrbracket$.

Nous pouvons donc simuler les automates bidirectionnels sur Γ^* par des automates sur Γ_2 avec tests dans $\mathcal{L}_2$.

Proposition 4.8.4. *Pour tout automate bidirectionnel A, il existe un automate B sur Γ_2 avec tests dans $\mathcal{L}_2$ de taille polynomiale en la taille A tel que $\mathcal{M}(B) = \mathcal{L}(A)$.*

Par le théorème 4.8.2 et comme les langages de $\mathcal{L}_2$ sont acceptés par des automates entièrement déterministes et complets de taille polynomiale en la taille de Γ, nous réobtenons le résultat de [She59].

Proposition 4.8.5 ([She59])**.** *Pour tout automate A bidirectionnel sur Γ^*, il existe un automate B fini déterministe et complet de taille bornée par* $\exp[1](|A|)$ *et acceptant $\mathcal{L}(A)$.*

Nous considérons naturellement la version alternante des automates bidirectionnels en nous basant sur la définition de [CKS81]. Il est important de remarquer que cette notion d'alternance est plus faible que celle considérée par exemple dans [LLS84, GGK91, GH96, EH99]. En effet, nous n'autorisons que des exécutions finies alors que ces auteurs autorisent des exécutions acceptantes infinies. Pour pouvoir simuler cette notion plus générale d'alternance, il faudrait définir une notion d'alternance plus générale pour les automates sur Γ_k (cf. remarque 4.3.17).

Définition 4.8.6 ([CKS81])**.** Un automate bidirectionnel alternant sur Γ^* est un uplet (Q,I,F,Δ) où Q est un ensemble fini d'états, I et $F \subseteq Q$ sont respectivement l'ensemble des états initiaux et finaux et où l'ensemble des transitions Δ est un sous-ensemble de $Q \times 2^{\Gamma \times Q \times \mathcal{I}}$.

Les notions d'exécution et d'acceptation sont définies de manière analogue à celles des automates alternants sur Γ_k. En utilisant l'encodage présenté précédem-

ment, nous pouvons construire pour tout automate A bidirectionnel alternant sur Γ^* un automate B alternant sur Γ_2 de taille polynomiale en la taille de A et tel que $\mathcal{M}(B) = \mathcal{L}(A)$. Remarquons que comme le montre la proposition 4.3.38, il n'est pas nécessaire de faire apparaître explicitement les tests de $\mathcal{L}_2$ dans l'automate B. Par le théorème 4.8.2, nous obtenons le résultat suivant.

Proposition 4.8.7 ([CKS81]). *Pour tout automate A bidirectionnel et alternant sur Γ^*, il existe un automate B fini déterministe et complet de taille bornée par* $\exp[2](|Q_A|)$ *et acceptant $\mathcal{L}(A)$.*

4.8.2 Automates à galets

Nous considérons maintenant l'ajout de galets aux automates bidirectionnels. Intuitivement, un automate bidirectionnel avec un galet dispose d'un marqueur qu'il peut déposer sur sa bande et qu'il peut reprendre si la tête de lecture est positionnée sur le marqueur et dont il peut tester la présence sous la tête de lecture.

Une configuration est un uplet (w,i,j,q) où $w \in \Gamma^+$, $i \in [1,|w|]$ désigne la position de la tête de lecture et $j \in [0,|w|]$ désigne la position du galet si $j \neq 0$ et son absence sinon. Nous ajoutons donc aux instructions de $\mathcal{I}$ des instructions Pose_1, Leve_1, Galet_1 et $\neg\text{Galet}_1$ pour obtenir l'ensemble des instructions $\mathcal{I}_1$. Les instructions de $\mathcal{I}$ sont étendues de manière naturelle en des fonctions partielles de $\{(w,i,j) \mid w \in \Gamma^+,\ i \in [1,|w|] \text{ et } j \in [0,|w|]\}$. L'instruction Pose_1 appliquée à $(w,i,0)$ donne (w,i,i) et est non définie sinon. L'instruction Leve_1 appliquée à (w,i,j) donne $(w,i,0)$ si $j = i$ et est non définie sinon. Les instructions Galet_1 et $\neg\text{Galet}_1$ ne modifie pas la configuration est sont respectivement définies si $i = j$ ou si $i \neq j$.

Les automates bidirectionnels (resp. et alternant) avec un galet sont définis en remplaçant $\mathcal{I}$ par $\mathcal{I}_1$ dans la définition 4.8.3 (resp. dans la définition 4.8.6).

Pour simuler la position du galet, nous ajoutons un niveau de pile et nous simulons un automate avec un galet par un automate sur Γ_3. Nous reprenons l'encodage présenté dans le sous-paragraphe précédent. Pour tout (w,i,j) avec $w \in \Gamma^+$, $i \in [1,|w|]$ et $j \in [0,|w|]$, nous définissons la pile de $\text{Stacks}_3(\Gamma)$ notée $[\![(w,i,j)]\!]$ par:

- $[[[w][w(1),\ldots,w(j)]][[w][w(1),\ldots,w(i)]]]_3$ si $j \neq 0$,
- $[[[w][w(1),\ldots,w(j)]]]_3$ sinon.

Une configuration (w,i,j,q) de l'automate bidirectionnel avec un galet est simulée par une configuration $(q,[\![(w,i,j)]\!])$ de l'automate sur Γ_3.

Avant de présenter l'encodage des instructions de $\mathcal{I}_1$, nous introduisons des langages de tests dans $\text{Rat}_3(\Gamma)$. Nous reprenons les langages de $\mathcal{L}_2$ vus comme des langages de piles de niveau 3 (*i.e.* $\{s \in \text{Stacks}_3(\Gamma) \mid \text{top}_2(s) \in L\}$ pour $L \in \mathcal{L}_2$) et

nous ajoutons les langages de $\mathrm{Rat}_3(\Gamma)$ suivants: $\mathrm{NPose} = \{[\, s \,]_3 \mid s \in \mathrm{Stacks}_2(\Gamma)\}$, $\mathrm{Galet} = \{[\, ss \,]_2 \mid s \in \mathrm{Stacks}_2(\Gamma)\}$ et leurs complémentaires notés respectivement $\mathrm{NPose}^{\mathrm{c}}$ et $\mathrm{Galet}^{\mathrm{c}}$.

Les instructions de $\mathcal{I}$ conservent les même simulations que dans le sous-paragraphe précédent. Pour les nouvelles instructions $\rho \in \mathcal{I}_1 \setminus \mathcal{I}$, nous définissons un ensemble non-ambigu R_ρ par:

$$\begin{array}{lcllcl} R_{\mathrm{Pose}} & = & \{\, \mathcal{T}_{\mathrm{NPose}} \cdot 2 \,\} & R_{\mathrm{Leve}} & = & \{\, \bar{2} \,\} \\ R_{\mathrm{Galet}} & = & \{\, \bar{2} \cdot 2 \,\} & R_{\neg\mathrm{Galet}} & = & \{\, \mathcal{T}_{\mathrm{Galet}^{\mathrm{c}}} \,\} \end{array}$$

Nous vérifions aisément que ces ensembles non-ambigus simulent bien les instructions correspondantes. Ainsi nous pouvons construire pour tout automate A bidirectionnel avec 1 galet un automate sur Γ_3 avec tests dans $\mathcal{L}_2$ de taille polynomiale en la taille de A et tel que $\mathcal{M}(B) = \mathcal{L}(A)$.

Proposition 4.8.8. *Pour tout automate bidirectionnel (resp. et alternant) avec* 1 *galet, il existe un automate déterministe et complet sur* Γ *acceptant le même langage et de taille bornée par* $\exp[2](|A|)$ *(resp.* $\exp[3](|Q_A|)$*).*

Pour les automates bidirectionnels avec 1 galet, ce résultat a été obtenu dans [BH67]. Pour une version plus générale d'alternance, un résultat similaire a été obtenu dans [GGK91].

Remarque 4.8.9. Par un encodage plus élaboré, nous pouvons autoriser l'automate à supprimer le galet sans que la tête de lecture soit positionnée sur le galet. Pour cela, il faut mémoriser dans l'état la présence du galet sur la bande. Enlever le galet consiste à simplement modifier l'état. Le nombre de piles de niveau 2 contenues dans la pile de niveau 3 n'est alors plus borné et la pile de niveau 3 contient l'historique de toutes les positions sur lesquelles ont été posées le galet.

Dans [GH96], les auteurs proposent un modèle d'automate bidirectionnel manipulant k galets avec $k \geq 2$ et acceptant uniquement les langages rationnels. En effet, sans restrictions sur la manipulation de 2 galets, il est aisé de construire un automate acceptant le langage algébrique $\{a^n b^n \mid n \geq 1\}$. Les auteurs de [GH96] introduisent une «discipline de pile» dans la manipulation des k galets. Cette condition se traduit par le fait que l'automate peut poser le $i^{\text{ème}}$ galet que si les $i-1$ premiers galets sont déjà sur la bande et ne peut enlever le $i^{\text{ème}}$ galet que si le $i+1^{\text{ème}}$ galet n'est plus sur la bande et que si la tête de lecture se trouve sur le $i^{\text{ème}}$ galet. Dans [GH96] des restrictions supplémentaires sont introduites qui sont levées dans [EH99].

En adaptant l'encodage présenté pour les automates, nous pouvons construire pour tout automate bidirectionnel avec k galets, un automate sur Γ_{k+2} équivalent de taille polynomiale. Grâce au théorème 4.8.2, nous dérivons les résultats de

complexité suivants qui étendent les résultats obtenus dans [GH96].

Proposition 4.8.10. *Pour tout automate bidirectionnel (resp. et alternant) avec k galets, il existe un automate déterministe et complet sur Γ acceptant le même langage et de taille bornée par* $\exp[k+1](|A|)$ *(resp.* $\exp[k+2](|Q_A|)$*).*

Chapitre 5

Graphes des automates à piles de piles

Dans ce chapitre, nous définissons et étudions les trois familles de graphes infinis associées aux automates à pile sur Ops_k: les graphes enracinés, les graphes des configurations et les graphes des transitions. Nous donnons plusieurs caractérisations internes et externes de ces graphes. En particulier, les résultats obtenus pour la classe des graphes des transitions d'automates à piles sur Ops_k sont résumées par le théorème 5.3.6 que nous rappelons ici.

Les propositions suivantes sont équivalentes à isomorphisme près:

1. G est un graphe des transitions d'un automates à piles sur Ops_k,
2. G est un graphe des transitions d'un automates à piles sur COps_k,
3. G est un graphe préfixe-reconnaissable de niveau k,
4. G est un obtenu en itérant k fois l'application d'une substitution rationnelle inverse suivi d'un dépliage en partant d'un graphe fini,
5. G est un obtenu en itérant k fois l'application d'une transduction monadique suivie d'une opération de Treegraph en partant d'un graphe fini,
6. G est interprétable dans $\mathrm{GStacks}_k$ (resp. $\mathrm{TStacks}_k$).

Une partie de ces résultats a été obtenue en collaboration avec Stefan Wöhrle et a été présentée [CW03] ainsi que dans sa thèse [Wöh05]. Les preuves, que nous donnons ici, s'appuient beaucoup sur la notion de rationalité développée dans le chapitre précédent et sont (si l'on admet les résultats sur la rationalité) plus simples.

5.1 Définitions et propriétés

Dans ce paragraphe, nous définissons les différents graphes associés aux automates à pile sur Ops_k. Dans le sous-paragraphe 4.1.5, nous avons défini les

systèmes de transitions étiquetés associés aux automates à pile sur Ops_k. Comme dans le sous-paragraphe 2.1.1.2, nous introduisons pour chaque niveau $k \geq 1$, trois classes de graphes: la classe des graphes enracinés (notée RHPDS_k), la classe des graphes des configurations (notée CHPDS_k) et la classe des graphes des transitions (notée GHPDS_k).

5.1.1 Les graphes enracinés

Soit $\mathcal{A} = (\Gamma,\Sigma,\tau,Q,I,F,\Delta)$ un automate à pile sur Ops_k et c_0 une configuration de cet automate, le graphe de $\mathcal{A}$ enraciné en c_0 est obtenu en restreignant le graphe de son LTS aux configurations accessibles depuis la configuration c_0.

Définition 5.1.1. La classe RHPDS_k est la classe des graphes isomorphes à un graphe enraciné d'automate à pile sur Ops_k.

Exemple 5.1.2. Reprenons l'automate à pile sur Ops_2 acceptant le langage $\{w\$w \mid w \in \{a,b\}^*\}$ présenté dans l'exemple 4.1.14. Son graphe enraciné en la configuration $(i, [\,]_2)$ est présenté dans la figure 5.1.

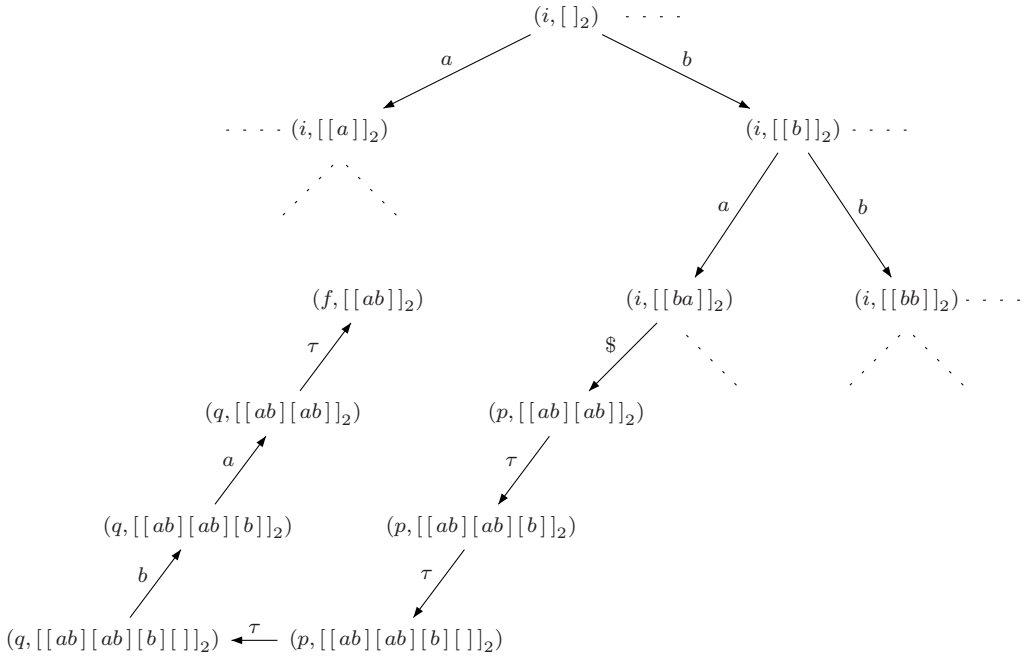


FIG. 5.1 – *Graphe enraciné en* $(i, [\,]_2)$ *de l'automate à pile de l'exemple 4.1.14.*

Remarque 5.1.3.

1. Si nous ne considérons que les graphes engendrés par les automates à pile sur Ops_k, nous pouvons omettre les états initiaux et les états finaux.
2. Tout graphe G dans RHPDS_k est isomorphe au graphe d'un automate sur Ops_k ne possèdant qu'un unique état et dont l'alphabet de pile ne contient que deux symboles. . Pour un tel automate, une configuration est entièrement décrite par une pile de niveau k. Ainsi lorsque nous considérerons des automates ne possédant qu'un seul état, nous supposerons qu'une configuration est simplement une pile de niveau k. Intuitivement pour se ramener à un automate ne possédant qu'un seul état, il suffit d'empiler l'état en haut de la pile. Une configuration (q,s) est donc représentée par la pile $\mathrm{push}_q(s)$ et une transition (p,θ,a,q) est alors remplacée par $(q_0,\mathrm{pop}_p\theta\mathrm{push}_q,q_0)$ où p_0 est l'unique état de notre automate. Pour passer d'un alphabet de pile Ξ de taille arbitraire à un alphabet Γ contenant deux symboles, il suffit d'encoder chaque symbole de Ξ par une suite de symboles de Γ de taille $\lceil \log_2(|\Gamma|) \rceil$.

Une propriété fondamentale de ces graphes est que si l'on ajoute les tests rationnels de niveau k aux opérations de Ops_k, la classe des graphes enracinés n'est pas enrichie. Cette propriété repose sur une version enrichie de l'encodage des piles de niveau k présenté dans le paragraphe 4.1.5.2 et sur la caractérisation des langages de Rat_k par des automates sur Γ_k avec tests dans Rat_{k-1} déterministes et complets (cf. théorème 4.4.15).

Proposition 5.1.4. *Les graphes enracinés des automates à pile sur* $\mathrm{Ops}_k \cup \{\mathrm{Test}_R \mid R \in Rat_k\}$ *appartiennent à* RHPDS_k.

Démonstration. Soit $\mathcal{A} = (\Gamma,\Sigma,\tau,Q_A,\Delta_A)$ un automate à pile sur $\mathrm{Ops}_{k_0} \cup \{\mathrm{Test}_R \mid R \in \mathrm{Rat}_{k_0}\}$ pour $k_0 \geq 1$.

Notons $\mathcal{L}_{k_0} = \{ L_1^{k_0}, \ldots, L_{n_{k_0}}^{k_0} \} \subseteq \mathrm{Rat}_{k_0}$, l'ensemble des éléments de Rat_{k_0} apparaissant dans $\mathcal{A}$. Il existe un automate $A_{k_0} = (Q_{k_0},\{ i_{k_0} \}, F_{k_0},\Delta_{k_0})$ déterministe et complet sur Γ_k avec tests dans Rat_{k_0-1} et une relation $\eta_{k_0} \subseteq Q_A \times [1,n_{k_0}]$ tels que pour toute pile $s \in \mathrm{Stacks}_{k_0}(\Gamma)$ et pour tout $i \in [1,n_{k_0}]$,

$$s \in L_i^{k_0} \Leftrightarrow (A_{k_0}(s),i) \in \eta_{k_0}.$$

Rappelons que nous notons $A_{k_0}(s)$ l'unique état de Q_{k_0} tel qu'il existe un calcul de A_{k_0} partant de la configuration $(i_{k_0},[\,]_{k_0})$ et arrivant dans la configuration $(A_{k_0}(s),s)$ (cf. lemme 4.4.3). L'automate A_{k_0} peut être construit en faisant par exemple le produit comme dans la proposition 4.4.17 des automates déterministes et complets avec tests dans Rat_{k_0-1} acceptant les langages de $\mathcal{L}_{k_0}$ (cf. corollaire 4.4.9).

Pour tout $\ell \in [1,k_0-1]$, nous notons $\mathcal{L}_\ell = \{ L_1^\ell, \ldots, L_{n_\ell}^\ell \}$ l'ensemble des éléments de Rat_ℓ apparaissant dans $\mathcal{A}_{\ell+1}$ et nous notons $A_\ell = (Q_\ell,\{ i_\ell \},F_\ell,\Delta_\ell)$ un automate

déterministe et complet avec tests dans $\mathrm{Rat}_{\ell-1}$ et une relation $\eta_\ell \subseteq Q_A \times [1,n_\ell]$ tels que pour toute pile $s \in \mathrm{Stacks}_\ell(\Gamma)$ et pour tout $i \in [1,n_\ell]$,

$$s \in L_i^\ell \Leftrightarrow (A_\ell(s),i) \in \eta_\ell.$$

Nous supposerons sans perte de généralité que les ensembles d'états des automates A_ℓ sont disjoints. Nous noterons Q l'union de tous les ensembles d'états des automates $\mathcal{A}_\ell$ (*i.e.* $Q = \bigcup_{\ell\in[1,k_0]} Q_\ell$).

Nous allons étendre l'encodage des piles de $\mathrm{Stacks}_k(\Gamma)$ présenté dans le paragraphe 4.1.5.2. Rappelons que dans cet encodage, une pile $s \in \mathrm{Stacks}_k(\Gamma)$ se voit associer une pile de $\mathrm{Stacks}_k(\Gamma_k^\circ)$ notée $[\![\, s \,]\!]$. Intuitivement cet encodage permet d'accéder à la suite réduite de s et aux suites réduites des piles $\mathrm{top}_\ell(s)$ pour $\ell \in [1,k_0-1]$. Nous enrichissons cet encodage pour rendre accessibles les états $A_{k_0}(s)$ et $A_\ell(\mathrm{top}_\ell(s))$ pour tout $\ell \in [1,k_0-1]$.

Formellement, l'encodage associe, à toute pile s de niveau $n \leq k_0$ sur Γ, une pile de niveau n sur $\Gamma_n^\circ \cup \bigcup_{j\in[1,n]} Q_j$ notée $[\![\, s \,]\!]$. Cet encodage est défini par récurrence sur le niveau $n \in [1,k]$.

Au niveau 1, l'encodage est défini par l'équation de récurrence suivante:

$$\begin{cases} [\![\, [\,]_1 \,]\!] &= [\, i_1 \,]_1 \\ [\![\, [\, sa \,]_1 \,]\!] &= \mathrm{push}_{A_1(sa)}(\mathrm{push}_a([\![\, s \,]\!]_1)). \end{cases}$$

Au niveau $n+1 \in [2,k]$, considérons une pile $s \neq [\,]_{n+1}$ avec sa suite réduite $\rho \in (\Gamma_n^\circ \cup \{\, n \,\})^*$ (cf. remarque 4.1.11). Pour tout $\ell \in [1,|\rho|]$, nous définissons $\rho_\ell = \rho(1)\dots\rho(\ell)$ et $\tilde{\rho}_\ell$ la suite obtenue en effaçant les occurrences de l'instruction k dans ρ_ℓ. La suite $\tilde{\rho}_\ell$ appartient à $(\Gamma_n^\circ)^*$. Enfin, nous définissons, pour tout $\ell \in [1,|\rho|]$, la pile $s_\ell = \mathcal{R}(\tilde{\rho}_\ell)([\,]_n)$, $r_\ell = \mathcal{R}(\rho_\ell)([\,]_n)$ et $q_\ell = A_{n+1}(r_\ell) \in Q_{n+1}$.

Nous pouvons maintenant définir $[\![\, \cdot \,]\!]$ pour les piles de niveau $n+1$.

$$\begin{array}{lcl} [\![\, [\,]_{n+1} \,]\!] &=& [\mathrm{push}_{i_{n+1}}[\![\, [\,]_k \,]\!]]_{n+1} \\ [\![\, s \,]\!] &=& [\, \mathrm{push}_{i_{n+1}}([\![\, [\,]_n \,]\!]), \dots, \mathrm{push}_{q_{|\rho|}}(\mathrm{push}_{\rho(|\rho|)}([\![\, s_{|\rho|} \,]\!]))\,]_{n+1} \end{array} \tag{5.1}$$

Comme dans le paragraphe 4.1.5.2, pour tout niveau $k \leq k_0$ et pour toute opération $\theta \in \mathrm{Ops}_k \cup \{\mathrm{Test}_L \mid L \in \mathcal{L}_\ell \text{ et } \ell \in [1,k]\}$, il existe un ensemble fini non-ambigu d'éléments de $\mathrm{Ops}_k^\star$, noté $[\![\, \theta \,]\!]_k$, tel que $[\![\, \theta \,]\!]_k([\![\, s \,]\!])$ est défini si et seulement si $\theta(s)$ l'est et dans ce cas, $[\![\, \theta \,]\!]_k([\![\, s \,]\!]) = [\![\, \theta(s) \,]\!]$.

La définition des ensembles $[\![\, \theta \,]\!]_k$ est une adaptation immédiate des encodages présentés dans le paragraphe 4.1.5.2. Une différence notable est que nous n'utilisons plus les opérations de destruction inconditionnelle destr_ℓ mais les opérations de destruction symétrique $\overline{\mathrm{copy}}_\ell$. Dans l'encodage précédent, l'opération destr_k était utilisée pour passer de l'encodage d'une pile $s \neq [\,]_{k+1}$ de suite réduite ρ à celui de la pile s' ayant pour suite réduite $\rho(1)\dots\rho(|\rho|-1)$. Ainsi l'on simule

l'application de l'opération $\theta = \mathcal{R}(\overline{\rho(\rho)})$ à la pile s. En effet, $[\![s']\!] = \mathrm{destr}_k([\![s]\!])$ et $s' = \theta(s)$.
Pour passer de $[\![s]\!]$ à $[\![s']\!]$ sans utiliser destr_k, il suffit de considérer l'ensemble fini $\mathcal{R}((\Gamma_{k-1}^{\circ})^{\leq 4k})\overline{k})$. En effet, on vérifie aisément que la suite réduite $\rho_{s,s'}$ est au plus de taille $4k+1$.

Nous définissons un automate $\mathcal{B} = (\Gamma,\Sigma,\tau,Q_A,I,F,\Delta_{\mathcal{B}})$ où l'ensemble des transitions $\Delta_{\mathcal{B}}$ est défini par:

$$\Delta_{\mathcal{B}} = \{(p,x,\theta',q) \mid (p,x,\theta,q) \in \Delta_{\mathcal{A}} \text{ et } \theta' \in [\![\theta]\!]_k\}.$$

Pour toutes piles s et $s' \in \mathrm{Stacks}_{k_0}(\Gamma)$ et $x \in \Sigma \cup \{ x \}$, il suit de ce qui précède que:

$$(p,s) \xrightarrow[\mathcal{A}]{x} (q,s') \;\Leftrightarrow\; (p,[\![s]\!]) \xrightarrow[\mathcal{B}]{x} (q,[\![s']\!]).$$

Pour toute configuration (p,s) de $\mathcal{A}$, le graphe de $\mathcal{A}$ enraciné en (p,s) est isomorphe au graphe de $\mathcal{B}$ enraciné en $(p,[\![s]\!])$. □

5.1.2 Les graphes des configurations

Le graphe des configurations d'un automate à pile $\mathcal{A} = (\Gamma,\Sigma,\tau,Q,\Delta)$ sur Ops_k est défini en restreignant le graphe de son LTS à un ensemble «rationnel» de configurations. Plus précisément, un ensemble rationnel de configurations est donné par une famille $(R_q)_{q\in Q}$ d'ensembles de $\mathrm{Rat}_k(\Gamma)$ et est égal à $\{(q,s) \in Q \times \mathrm{Stacks}_k(\Gamma) \mid s \in R_q\}$. Pour les automates à pile sur Ops_k de la forme présentée dans la remarque 5.1.3, un ensemble rationnel de configurations est simplement un ensemble rationnel de piles.

Définition 5.1.5. La classe CHPDS_k est la classe des graphes isomorphes à un graphe des configurations d'un automate à pile sur Ops_k.

Exemple 5.1.6. Considérons l'alphabet de pile $\Gamma = \{ A \}$, l'alphabet d'entrée $\Sigma = \{ a,b \}$ et l'automate $\mathcal{A} = (\Gamma,\Sigma,\tau,Q,\Delta)$ à pile sur Ops_2 avec $Q = \{ q_0,q_1 \}$ et

$$\Delta = \{ (q_0,\mathrm{push}_A,a,q_0),(q_1,\mathrm{push}_A,b,q_1),(q_1,\mathrm{push}_A \cdot \overline{\mathrm{copy}}_1,b,q_0) \}.$$

Son graphe des configurations pour la restriction:

- $R_{q_0} = \{[\,[\,A^n\,]\,]_2 \mid n \geq 0\} = \mathcal{R}(A^*)([\,])$,
- $R_{q_1} = \{[\,[\,A^n\,]\,[\,A^m\,]\,] \mid n \geq 1 \text{ et } m < n\} = \mathcal{R}(A^+1\bar{A}^+)([\,]_2)$

est donné dans la figure 5.2. Comme ce graphe n'admet pas de racine (*i.e.* un sommet permettant d'accéder à tous les autres sommets), il n'appartient pas à RHPDS_2.

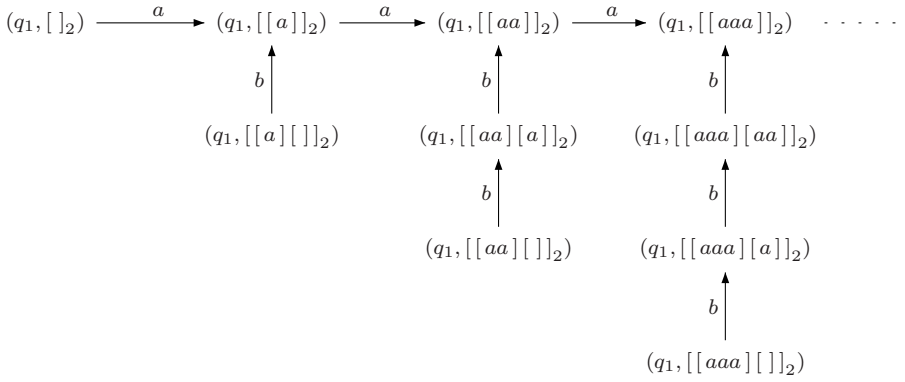


FIG. 5.2 – *Graphes des configurations de l'automate à pile* $\mathcal{A}$ *sur* Ops_2.

Remarque 5.1.7. Comme pour les graphes enracinés (cf. remarque 5.1.3), tout graphe G dans CHPDS_k est isomorphe à un graphe des configurations d'un automate à pile sur Ops_k avec un unique état et un alphabet réduit à deux symboles.

Les graphes enracinés des automates à pile sur Ops_k sont des exemples de graphes des configurations d'automates à pile sur Ops_k. Considérons un automate à pile $\mathcal{A}$ sur Ops_k et une configuration c_0 de cet automate. Pour tout $q \in Q$, l'ensemble des piles s de niveau k, noté R_q, telles que la configuration (q,s) soit accessible depuis c_0 est un ensemble rationnel (*i.e.* $R_q = \{s \in \mathrm{Stacks}_k \mid c_0 \Longrightarrow (q,s)\} \in \mathrm{Rat}_k$). Le graphe enraciné de $\mathcal{A}$ en c_0 est donc le graphe des configurations de $\mathcal{A}$ pour la restriction $(R_q)_{q\in Q}$.

Proposition 5.1.8. *Pour tout* $k \geq 1$, $\mathrm{RHPDS}_k \subsetneq \mathrm{CHPDS}_k$.

Exemple 5.1.9. Reprenons le graphe enraciné de l'automate à pile $\mathcal{A}$ présenté dans l'exemple 4.1.14. Son graphe enraciné en $(q_0,[\,]_2)$ est présenté dans la figure 5.1. Ce graphe est égal aux graphes des configurations de $\mathcal{A}$ pour la restriction $(R_q)_{q\in Q}$ où:

- $R_i = R_f = \mathcal{R}(\{a,b\}^*)([\,]_2)$,
- $R_p = R_q = \mathcal{R}(\{a,b\}^*1(\{\bar{a},\bar{b}\}1)^*\perp_1)([\,]_2)$.

Comme nous l'avons vu dans l'exemple 5.2, l'inclusion de la classe des graphes enracinés dans la classe des graphes des configurations est stricte. Cependant la classe RHPDS_k se caractérise de manière naturelle dans la classe CHPDS_k.

Proposition 5.1.10. *Un graphe* G *de* CHPDS_k *appartient à* RHPDS_k *si et seulement si* G *admet au moins une racine.*

Démonstration. Seule l'implication réciproque présente une difficulté. Soit $G \in$

CHPDS_k et admettant une racine $r \in V_G$. Par définition de CHPDS_k et par la remarque 5.1.7, il existe un automate à pile $\mathcal{A} = (\Gamma,\Sigma,\tau,\{\, q_0 \,\},\Delta_{\mathcal{A}})$ et un ensemble rationnel R de piles de niveau k tels que G soit isomorphisme au graphe H des configurations de $\mathcal{A}$ pour la restriction R. Le graphe H admet une racine $s_0 \in R$.

Considérons l'automate à pile $\mathcal{B} = (\Gamma,\Sigma,\tau,\{\, q_0 \,\},\Delta_{\mathcal{B}})$ sur $\mathrm{Ops}_k \cup \{\, \mathrm{Test}_R \,\}$ dont l'ensemble des transitions $\Delta_{\mathcal{B}}$ est donné par:

$$\{(q_0,\theta \cdot \mathrm{Test}_R,q_0) \mid (q_0,\theta,q_0) \in \Delta_A\}.$$

Il est facile de vérifier que le graphe de $\mathcal{B}$ enraciné en s_0 est égal à H. Par la proposition 5.1.4, il découle que H (et donc G) appartient à RHPDS_k. □

Comme pour la classe RHPDS_k, l'ajout des opérations de tests rationnels de niveau k ne modifie la classe CHPDS_k.

Proposition 5.1.11. *Les graphes des configurations des automates à pile sur* $\mathrm{Ops}_k \cup \{\mathrm{Test}_R \mid R \in Rat_k\}$ *appartiennent à* CHPDS_k.

Démonstration. Soit $\mathcal{A} = (\Sigma,\Gamma,\tau,Q_{\mathcal{A}},\Delta_{\mathcal{A}})$ un automate à pile sur $\mathrm{Ops}_k \cup \{\mathrm{Test}_R \mid R \in \mathrm{Rat}_k\}$ et R un ensemble rationnel de configurations donné par une famille $(R_p)_{p \in Q_{\mathcal{A}}}$ d'ensembles de $\mathrm{Rat}_k(\Gamma)$. Soit $\mathcal{L} = \{\, L_1,\ldots,L_n \,\} \subset \mathrm{Rat}_k(\Gamma)$ l'ensemble des langages de tests utilisés dans $\mathcal{A}$.

Nous allons montrer que le graphe des configurations de $\mathcal{A}$ pour la restriction à R, noté G, appartient à CHPDS_k. Pour cela, nous construisons un automate à pile $\mathcal{B}$ sur Ops_k et un ensemble rationnel de configurations S donné par une famille $(S_q)_{q \in Q}$ tels que le graphe des configurations de $\mathcal{B}$ pour la restriction à S, noté H, soit isomorphe au graphe G.

Pour cela, nous reprenons l'encodage $[\![\cdot]\!]$ des piles de niveau k présenté dans la proposition 5.1.4 ainsi que l'automate $\mathcal{B}$ qui y est construit. Il suffit de remarquer que pour tout ensemble rationnel $R \in \mathrm{Rat}_k(\Gamma)$, l'ensemble $[\![R]\!]$ des encodages des piles de R est lui aussi rationnel. Il suit alors que le graphe des configurations $\mathcal{B}$ pour la restriction $([\![R_q]\!])_{q \in Q_{\mathcal{A}}}$ est isomorphe au graphe G. Donc le graphe G appartient à CHPDS. □

5.1.3 Les graphes de transitions

Pour définir la classe des graphes des transitions des automates à pile sur Ops_k, il nous faut nous restreindre à des automates à pile sur Ops_k normalisés (cf. paragraphe 2.1.1.2). La restriction aux automates à pile normalisés sur Ops_k ne change pas la famille des langages acceptés.

Proposition 5.1.12. *Pour tout langage L indexé de niveau k, il existe un automate à pile sur* Ops_k *normalisé acceptant L.*

Démonstration. Soit $L \subseteq \Sigma^*$ un langage indexé de niveau k. Par le théorème 4.1.23, il existe un automate à pile $\mathcal{A} = (\Sigma,\Gamma,\tau,Q_A,I_A,F_A,\Delta_A)$ sur Ops_k acceptant L. Nous construisons un automate à pile $\mathcal{B} = (\Sigma,\Gamma,\tau,Q_B,I_B,F_B,\Delta_B)$ normalisé acceptant L. Prenons $Q_B = Q_A^\tau \cup Q_A^\Sigma$ où Q_A^τ et Q_A^Σ sont deux ensembles disjoints de Q_A et en bijection avec Q_A. Pour tout $q \in Q_A$, nous noterons q^τ (resp. q^Σ) l'état correspondant de Q_A^τ (resp. Q_A^Σ).De plus, nous supposons que Q_A^τ et Q_A^Σ sont disjoints. Les ensembles d'états initiaux et finaux I_B et F_B sont respectivement égaux à I_A^τ et F_A^τ. L'ensemble des transitions Δ_B est donné par:

$$\begin{array}{rcl}\Delta_B & = & \{(p^\tau,\tau,\theta,q^\tau) \mid (p,\tau,\theta,q) \in \Delta_A\} \\ & \cup & \{(p^\tau,\tau,\mathrm{Id}_k,p^\Sigma) \mid p \in Q_A\} \\ & \cup & \{(p^\Sigma,a,\theta,q^\tau) \mid (p,a,\theta,q) \in \Delta_A \text{ et } a \in \Sigma\}.\end{array}$$

On vérifie aisément que $\mathcal{A}$ et $\mathcal{B}$ acceptent le même langage. De plus par construction, $\mathcal{B}$ est tel qu'une configuration ne peut pas être à la fois la source d'une transition étiquetée par τ et d'une transition étiquetée par Σ. Il suit donc que $\mathcal{B}$ est normalisé. □

Remarque 5.1.13. Nous pouvons obtenir un résultat plus précis : tout langage indexé de niveau k est accepté par un automate à pile sur Ops_k normalisé à partir d'une unique configuration initiale observable jusqu'à une unique configuration finale observable.

Proposition 5.1.14. *Pour tout automate à pile normalisé $\mathcal{A}$ sur Ops_k et pour tout ensemble rationnel de configurations R, les configurations observables (resp. internes) du graphe des configurations de $\mathcal{A}$ pour la restriction à R forment un ensemble rationnel de configurations.*

Démonstration. Soit $\mathcal{A} = (\Gamma,\Sigma,\tau,Q,\Delta_\mathcal{A})$ un automate à pile normalisé sur Ops_k et R un ensemble rationnel de configurations donné par une famille $(R_q)_{q\in Q}$. Nous noterons G le graphe des configurations de $\mathcal{A}$ pour la restriction R. Soit $p \in Q$, nous allons montrer que l'ensemble, noté O_p, des piles $s \in \mathrm{Stacks}_k$ telles que (q,s) soit une configuration observable est un ensemble rationnel. Le cas des configurations internes est similiaire. Il existe deux relations préfixe-reconnaissables de niveau k respectivement notées P^τ et P^σ telles que toutes piles s et $s' \in \mathrm{Stacks}_k(\Gamma)$,

$$\begin{cases} (s,s') \in P^\sigma & \Leftrightarrow \quad \exists p \in Q, (q,s) \xrightarrow[G]{a} (p,s') \text{ pour } a \in \Sigma, \\ (s,s') \in P^\tau & \Leftrightarrow \quad \exists p \in Q, (q,s) \xrightarrow[G]{a} (p,s'). \end{cases}$$

La relation est P^σ est définie par l'ensemble $\{\mathrm{Test}_{R_q} \cdot \theta_p \cdot \mathrm{Test}_{R_p} \mid (q,\theta,a,p) \in \Delta_\mathcal{A} \text{ et } x \in \Sigma\}$ et P^τ est définie par $\{\mathrm{Test}_{R_q} \cdot \theta_p \cdot \mathrm{Test}_{R_p} \mid (q,\theta,\tau,p) \in \Delta_\mathcal{A}\}$.

Il est aisé de vérifier que:

$$O_s = \mathrm{Dom}(P^\sigma) \cup (\mathrm{Stacks}_k(\Gamma) \setminus \mathrm{Dom}(P^\tau)) \cap (\mathrm{Stacks}_k(\Gamma) \setminus \mathrm{Dom}(P^\sigma)).$$

Par la proposition 4.5.13, $\mathrm{Dom}(P^\sigma)$ et $\mathrm{Dom}(P^\tau)$ sont des ensembles de Rat_k. Comme Rat_k est une algèbre de Boole, O_s appartient à Rat_k.

□

Le graphe des transitions d'un automate à pile normalisé sur Ops_k est obtenu en effectuant la τ-fermeture d'un de ses graphes des configurations.

Exemple 5.1.15. L'automate sur Ops_2 présenté dans l'exemple 4.1.14 est normalisé. Son graphe des transitions obtenu comme la τ-fermeture de son graphe des configurations présenté dans la figure 5.1 est donné dans la figure 5.3.

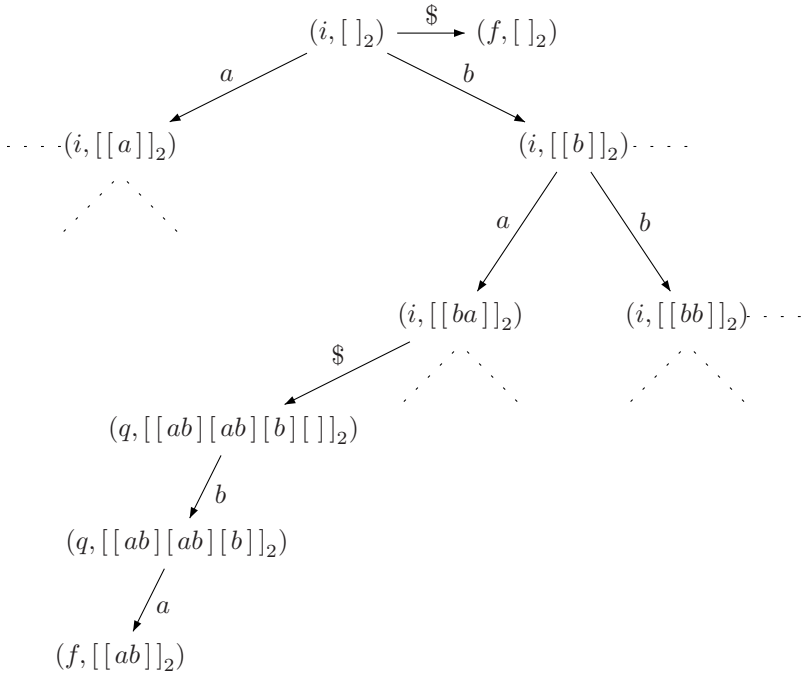


FIG. 5.3 – *Un graphe des transitions de l'automate à pile sur* Ops_2 *de l'exemple 4.3.3.*

Définition 5.1.16. La classe GHPDS_k est la classe des graphes isomorphes à un graphe des transitions d'un automate à pile sur Ops_k.

La classe des graphes des transitions GHPDS_k contient la classe des graphes des configurations. Cette inclusion est stricte car les graphes des transitions peuvent avoir un degré entrant et/ou sortant infini. Comme nous l'avons déjà mentionné au niveau 1, les graphes des configurations se caractérisent de manière très naturelle à l'intérieur des graphes des transitions. En effet au niveau 1, les

graphes de CHPDS_1 sont les graphes de GHPDS_1 de degré borné. Nous conjecturons qu'à partir du niveau 2, cette caractérisation n'est plus valable et que par exemple, le graphe des transitions présenté dans la figure 5.3 n'appartient pas à CHPDS_2.

Comme le montre la proposition ci-dessous, nous pouvons définir la classe des graphes de transitions en ne considérant que la τ-fermeture des graphes enracinés des automates à pile sur Ops_k.

Proposition 5.1.17. *Tout graphe G dans GHPDS_k est isomorphe à la τ-fermeture d'un graphe normalisé de RHPDS_k.*

Démonstration. Soit G un graphe dans GHPDS_k. Par définition de G et par la remarque 5.1.3, il existe un automate à pile $\mathcal{A} = (\Gamma,\Sigma,\tau,\{ q_0 \},\Delta_{\mathcal{A}})$ sur Ops_k et un ensemble rationnel de configurations [1] $R \in \mathrm{Rat}_k(\Gamma)$ tels que G soit isomorphe à la τ-fermeture du graphe des configurations, noté H, de l'automate $\mathcal{A}$ pour la restriction à R. Par la proposition 5.1.14, l'ensemble des configurations observables O est un ensemble rationnel de piles de niveau k.

Par définition de Rat_k, il existe un automate $A = (Q,\{i\},F,\Delta)$ fini étiqueté Γ_k tel que $\mathcal{S}(A) = O$. Nous pouvons sans perte de généralité supposer que q_0 n'appartient pas à Q. Considérons l'automate à pile $\mathcal{B} = (\Gamma,\Sigma,\tau,Q \cup \{ q_0 \},\Delta_B)$ sur $\mathrm{Ops}_k \cup \{\mathrm{Test}_R \mid R \in \mathrm{Rat}_k\}$ dont l'ensemble des transitions est donné par:

$$\begin{array}{rcl} \Delta_{\mathcal{B}} & = & \{(p,\mathcal{R}(\rho),\tau,q) \mid (p,\rho,q) \in \Delta\} \\ & \cup & \{(f,\mathrm{Id}_k,\tau,q_0) \mid (f \in F\} \\ & \cup & \{(q_0,\mathrm{Test}_R \cdot \theta \cdot \mathrm{Test}_R,x,q_0) \mid (q_0,\theta,x,q_0) \in \Delta_{\mathcal{A}}\}. \end{array}$$

Notons K le graphe $\mathcal{B}$ enraciné en $(i_0,[\ ]_k)$. Il est aisé de vérifier que pour toutes configurations observables $c_1 = (p,s)$ et $c_2 = (q,s')$ de K et pour tout $a \in \Sigma$, nous avons:

$$c_1 \xrightarrow[K]{a\tau^*} c_2 \Leftrightarrow c_1 \xrightarrow[H]{a\tau^*} c_2$$

Il découle donc que la τ-fermeture de H est égale à la τ-fermeture du graphe K. Le graphe G est bien donc la τ fermeture d'un graphe dans RHPDS_k. □

En utilisant les résultats du chapitre 4, nous pouvons étendre à tout niveau l'équivalence (à isomorphisme près) entre les graphes des transitions des automates à piles et les graphes préfixe-reconnaissables obtenue par Stirling dans [Sti00] et rappelée dans le théorème 2.2.6. Nous définissons naturellement la classe des graphes préfixe-reconnaissables de niveau k comme la classe des graphes de

1. Rappelons que comme l'automate $\mathcal{A}$ ne possède qu'un seul état, une configuration est simplement une pile.

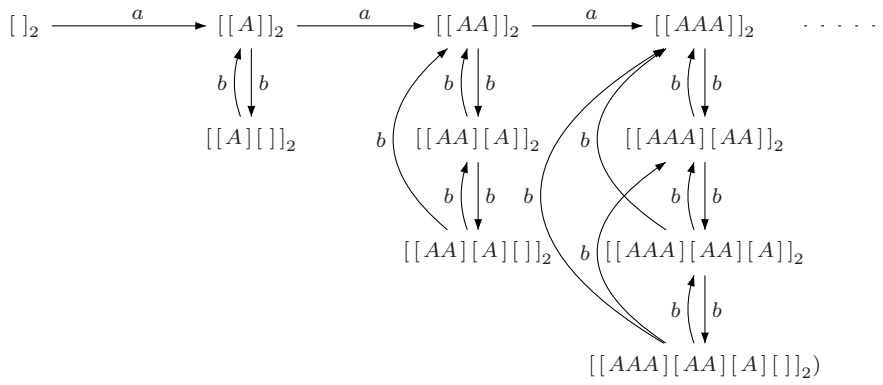


FIG. 5.4 – *Graphe préfixe-reconnaissable de niveau* 2.

calcul des relations préfixe-reconnaissables de niveau k définies dans le paragraphe 4.5.

Définition 5.1.18. Un graphe préfixe-reconnaissable G de niveau k étiqueté par $\Sigma \subset \Lambda$ est donné par une famille $(P_a)_{a\in\Sigma}$ de relations dans $\mathrm{PR}_k(\Gamma)$. Ce graphe a pour sommets des piles de $\mathrm{Stacks}_k(\Gamma)$ et est défini par:

$$G = \{(s,a,s') \mid (s,s') \in P_a\}.$$

Exemple 5.1.19. Considérons les deux relations préfixes P_a et P_b sur $\mathrm{Stacks}_2(\Gamma)$, avec Γ réduit au singleton $\{\, A \,\}$, définies par:

$$\left\{ \begin{array}{lcl} P_a & = & \mathcal{D}(T_{R_1} \cdot A) \\ P_b & = & \mathcal{D}(T_{R_2} \cdot 1\bar{A}) \cup \mathcal{D}(T_{R_2} \cdot (A\bar{1})^+). \end{array} \right.$$

où R_1 et R_2 sont respectivement les ensembles rationnels de piles de niveau 2 égaux à $\{[\,[\,A^n\,]\,]_2 \mid n \geq 0\} = \mathcal{R}(A^*)([\,]_2)$ et $\{\big[\,[\,A^n\,]\,[\,A^{n-1}\,]\ldots\big[\,A^{n-k}\,\big]\,\big]_2 \mid n \geq 0$ et $k \leq n\} = \mathcal{R}(A^+(1\bar{A})^*)([\,]_2)$. Le graphe préfixe-reconnaissable défini par $(P_x)_{x\in\{\,a,b\,\}}$ est donné dans la figure 5.4.

Théorème 5.1.20. *Un graphe G appartient à* GHPDS *si et seulement si il est isomorphe à un graphe préfixe-reconnaissable de niveau k.*

Démonstration. Nous prouvons les deux implications séparément.

Pour l'implication directe, considérons un graphe G appartenant à GHPDS_k. Par définition de GHPDS_k et par la remarque 5.1.7, il existe un automate à pile $\mathcal{A} = (\Sigma,\Gamma,\tau,\{\, q_0 \,\},\Delta)$ et un ensemble rationnel $R \in \mathrm{Rat}_k(\Gamma)$ tel que G soit isomorphe à la τ-fermeture du graphe des configurations de $\mathcal{A}$, noté H, pour la restriction R. Il est aisé de vérifier que pour tout $x \in \Sigma \cup \{\, \tau \,\}$, la relation

$P_x = \{(s,s') \in R \times R \mid (q_0,s) \xrightarrow[\mathcal{A}]{x} (q_0,s')\}$ est préfixe-reconnaissable de niveau k. Par la proposition 5.1.14, l'ensemble O des configurations observables du graphe H est un ensemble rationnel de piles de niveau k.

Par définition de la τ-fermeture, le graphe G est égal à:

$$\{(s,a,s') \mid (s,s') \in \mathrm{Test}_O \cdot P_a \cdot P_\tau^* \cdot \mathrm{Test}_O \text{ et } a \in \Sigma\}.$$

D'après les propriétés de fermeture des relations de PR_k présentées dans le théorème 4.5.16, les relations $\mathrm{Test}_O \cdot P_a \cdot P_\tau^* \cdot \mathrm{Test}_O$ appartiennent à PR_k. Il suit donc que G est un graphe préfixe-reconnaissable de niveau k.

Pour l'implication réciproque, considérons un graphe préfixe-reconnaissable G de niveau k donné par une famille $(P_a)_{a\in\Sigma}$ de relations préfixe-reconnaissables dans PR_k. Par définition de PR_k, il existe pour tout $a \in \Sigma$ un automate fini $A_a = (Q_a,\{ i_a \},\{ f_a \},\Delta_a)$ étiqueté par $\Gamma_k \cup \mathcal{T}_{\mathrm{Rat}_k}$ tel que $\mathcal{D}(A_a) = \mathrm{PR}_a$. Nous supposerons sans perte de généralité (quitte à utiliser des transitions étiquetées par Id_k) que i_a n'est destination d'aucune transition de A_a et que f_a n'est source d'aucune transition de A_a. De plus, nous supposerons que les ensembles d'états de ces automates sont deux à deux disjoints. Enfin, notons R le langage rationnel $\bigcup_{a\in\Sigma} \mathrm{Dom}(P_a) \cup \mathrm{Im}(P_a)$.

Nous définissons un automate à pile $\mathcal{B} = (\Gamma,\Sigma,\tau,Q_\mathcal{B},\Delta_\mathcal{B})$ sur $\mathrm{Ops}_k \cup \{\mathrm{Test}_R \mid R \in \mathrm{Rat}_k\}$. L'ensemble des états $Q_\mathcal{B}$ est égal à $\bigcup_{a\in\Sigma} Q_a \cup \{ q_0 \}$ et l'ensemble des transitions $\Delta_\mathcal{B}$ est défini par:

$$\begin{array}{rcl}
\Delta_\mathcal{B} & = & \{(p,\mathcal{R}(\gamma),\tau,q) \mid (p,\gamma,q) \in \Delta_a \text{ et } a \in \Sigma\} \\
& \cup & \{(p,\mathrm{Id}_k,\tau,p) \mid p \in Q_a \text{ et } a \in \Sigma\} \\
& \cup & \{(q_0,\mathrm{Test}_{\mathrm{Dom}(a)},a,i_a) \mid a \in \Sigma\} \\
& \cup & \{(f_a,\mathrm{Test}_{\mathrm{Im}(a)},a,q_0) \mid a \in \Sigma\}.
\end{array}$$

Considérons le graphe des configurations de $\mathcal{B}$, noté H, restreint à l'ensemble rationnel de configurations défini par $(R_q)_{q\in Q_\mathcal{B}}$ où $R_{q_0} = R$ et pour tout $q \in Q_\mathcal{B} \backslash \{ q_0 \}$, $R_q = \mathrm{Stacks}_k(\Gamma)$. Par construction, l'automate $\mathcal{B}$ est normalisé. Dans le graphe H, les configurations observables sont $\{(q_0,s) \mid s \in R\}$ et ses configurations internes sont $\{(q,s) \mid q \in Q_\mathcal{B} \setminus \{ q_0 \} \text{ et } s \in \mathrm{Stacks}_k(\Gamma)\}$. On vérifie aisément que pour tout s et $s' \in \mathrm{Stacks}_k(\Gamma)$,

$$(q_0,s) \xrightarrow[H]{a\tau^*} (q_0,s') \Leftrightarrow (s,s') \in P_a.$$

Il découle donc que la τ-fermeture de H est isomorphe au graphe G. □

5.1.4 Liens avec les graphes des automates à piles sur COps_k.

Dans ce sous-paragraphe, nous présentons quelques liens entre les graphes associés aux automates à pile sur le jeu d'opérations symétriques Ops_k et sur le

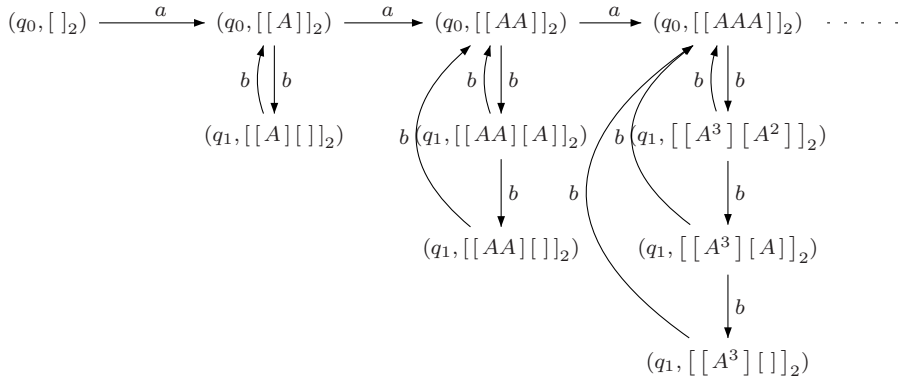


FIG. 5.5 – *Graphe enraciné d'un automate à pile sur* COps_2.

---

jeu d'opérations classiques COps_k. Nous noterons $\mathrm{RHPDS}_k^{\mathrm{c}}$ la classe des graphes enracinés des automates sur COps_k. Faute d'une notion pertinente de rationalité induite par les opérations de COps_k (cf. paragraphe 4.6), nous ne proposons pas de notion de graphes des configurations. Nous définissons les graphes des transitions comme la τ-fermeture des graphes enracinés. Nous noterons $\mathrm{GHPDS}_k^{\mathrm{c}}$ la classe des graphes des transitions des automates sur COps_k.

Une conséquence immédiate des constructions présentées dans le paragraphe 4.1.5.2 est que les graphes enracinés des automates sur Ops_k sont isomorphes à des graphes enracinés d'automates sur COps_k.

Théorème 5.1.21. *Pour tout* $k \geq 1$, $\mathrm{RHPDS}_k \subsetneq \mathrm{RHPDS}_k^{\mathrm{c}}$.

Démonstration. Soit G un graphe appartenant à RHPDS_k. Il existe un automate à pile $\mathcal{A}$ sur Ops_k et une configuration (q_0,s_0) de cet automate tels que G soit isomorphe au graphe enraciné de $\mathcal{A}$ en (q_0,s_0). Reprenons l'automate $\mathcal{B}$ sur COps_k construit dans la proposition 4.1.22. Le graphe de l'automate $\mathcal{B}$ enraciné en $(q_0,[\![\, s_0 \,]\!])$ est isomorphe au graphe G et donc G appartient à $\mathrm{RHPDS}_k^{\mathrm{c}}$. □

L'inclusion réciproque est fausse dès le niveau 2. En effet à cause des opérations de destruction inconditionnelle destr_k, les graphes enracinés des automates sur COps_k peuvent avoir un degré entrant non borné. Considérons par exemple, l'automate à pile $\mathcal{A} = (\{\, A \,\},\{\, a,b \,\},\{\, q_0,q_1 \,\},\Delta_{\mathcal{A}})$ sur COps_2 où l'ensemble des transitions $\Delta_{\mathcal{A}}$ est donné par:

$$\{\, (q_0,\mathrm{push}_A,a,q_0)\,,\,(q_0,\mathrm{copy}_1 \cdot \mathrm{pop}_A,b,q_1)\,,\,(q_1,\mathrm{pop}_A,b,q_1)\,,\,(q_1,\mathrm{destr}_1,b,q_0) \,\}.$$

Le graphe de $\mathcal{A}$ enraciné en $(q_0,[\,]_2)$ est présenté dans la figure 5.5. Comme le degré entrant de ce graphe est non borné, il n'appartient pas RHPDS_2.

Bien que la classe $\mathrm{RHPDS}_k^{\mathrm{c}}$ ne soit pas incluse dans la classe RHPDS_k, les graphes de $\mathrm{RHPDS}_k^{\mathrm{c}}$ sont isomorphes à des graphes de transitions d'automates sur Ops_k.

Proposition 5.1.22. *Pour tout $k \geq 1$, $\mathrm{RHPDS}_k^{\mathrm{c}} \subsetneq \mathrm{GHPDS}_k$.*

Démonstration. Soit G un graphe appartenant à $\mathrm{RHPDS}_k^{\mathrm{c}}$. Par définition de $\mathrm{RHPDS}_k^{\mathrm{c}}$ et par un raisonnement similaire à celui de la remarque 5.1.3, il existe un automate à pile $\mathcal{A} = (\Gamma,\Sigma,\tau,\{ q_0 \},\Delta_{\mathcal{A}})$ sur COps_k et une configuration $s_0 \in \mathrm{Stacks}_k(\Gamma)$ de cet automate tels que le graphe de l'automate $\mathcal{A}$ enraciné en s_0, noté H, soit isomorphe au graphe G. Il est aisé de vérifier que H est un graphe préfixe reconnaissable de niveau k. En effet pour toute opération $\theta \in \mathrm{COps}_k$, la relation sur les piles de $\mathrm{Stacks}_k(\Gamma)$, notée P_θ, induite par θ est une relation préfixe-reconnaissable de niveau k. De plus comme nous l'avons remarqué dans le paragraphe 4.6, l'ensemble des sommets de H est un ensemble de CRat_k et donc de Rat_k.
Donc pour tout $a \in \Sigma \cup \{ \tau \}$, considérons relation P_a définie par:

$$P_a = \bigcup_{(q_0,\theta_1 \ldots \theta_n,a,q_0) \in \Delta_a} \mathrm{Test}_R \cdot P_{\theta_1} \cdots P_{\theta_n} \cdot \mathrm{Test}_R$$

avec $\theta_1, \ldots, \theta_n \in \mathrm{COps}_k$. Pour tout $a \in \Sigma \cup \{ \tau \}$, la relation P_a appartient donc à PR_k et satisfait pour tout $(s,s') \in \mathrm{Stacks}_k(\Gamma) \times \mathrm{Stacks}_k(\Gamma)$, $(s,s') \in P_a$ si et seulement si $s \xrightarrow[H]{a} s'$. Le graphe H est donc égal au graphe préfixe de niveau k défini par $(P_a)_{a \in \Sigma}$. D'après le théorème 5.1.20, H (et donc G) appartient à GHPDS_k.

Le caractère strict de l'inclusion découle du fait que les graphes de RHPDS_k ont nécessairement un degré sortant borné alors que les graphes de GHPDS_k peuvent avoir un degré sortant non borné. Un exemple de graphe de GHPDS_2 de degré non borné est donné dans l'exemple 5.1.19. □

Les deux classes des graphes des transitions GHPDS_k et $\mathrm{GHPDS}_k^{\mathrm{c}}$ sont égales.

Théorème 5.1.23. *Pour tout $k \geq 1$, $\mathrm{GHPDS}_k = \mathrm{GHPDS}_k^{\mathrm{c}}$.*

Démonstration. Soit $k \geq 1$. L'inclusion $\mathrm{GHPDS}_k \subseteq \mathrm{GHPDS}_k^{\mathrm{c}}$ découle de l'inclusion $\mathrm{RHPDS}_k \subseteq \mathrm{RHPDS}_k^{\mathrm{c}}$ établie dans le théorème 5.1.21. Pour l'inclusion réciproque, considérons un graphe G appartenant à $\mathrm{GHPDS}_k^{\mathrm{c}}$. Par définition de $\mathrm{GHPDS}_k^{\mathrm{c}}$, le graphe G est isomorphe à la τ-fermeture d'un graphe H appartenant à $\mathrm{RHPDS}_k^{\mathrm{c}}$. Par la proposition 5.1.22, H appartient à GHPDS_k. D'après le théorème 5.1.20 et les propriétés de fermeture des relations de PR_k données dans le théorème 4.5.12, la τ-fermeture de H appartient elle aussi à GHPDS_k. □

5.2 Caractérisation par transformations de graphes

Dans ce paragraphe, nous établissons une caractérisation externe par transformations de graphes des classes CHPDS_k et GHPDS_k. Cette approche a été initiée par Caucal dans [Cau98, Cau02] où il introduit une hiérarchie de classes de graphes et d'arbres ayant une théorie monadique décidable. Ces classes de graphes et d'arbres sont définies en itérant le dépliage et les applications rationnelles inverses. Le point de départ est la classe Tree_0 des arbres finis. Pour tout $n \geq 0$, on a ensuite:

$$\mathrm{Graph}_k := \big\{G \mid G \approx h^{-1}(T), T \in \mathrm{Tree}_k \text{ et } h^{-1}\text{application rat. inv.}\big\},$$
$$\mathrm{Tree}_{k+1} := \big\{T \mid T \approx \mathrm{Unf}(G,r), G \in \mathrm{Graph}_k \text{ et } r \in V_G\big\}.$$

La classe Graph_0 est la classe des graphes finis. La classe Tree_1 est la classe des arbres réguliers c'est-à-dire la classe des arbres ayant uniquement un nombre fini de sous-arbres non isomorphes. La classe Graph_1 est la classe des graphes isomorphes à un graphe préfixe-reconnaissable (cf. théorème 2.2.6). Dans [CW03], nous avons montré en collaboration avec Stefan Wöhrle que pour tout $k \geq 1$, la classe Graph_k est en fait la classe des graphes des transitions des automates à piles sur Ops_k ou bien de manière équivalente la classe des graphes isomorphes à un graphe préfixe-reconnaissable de niveau k.

Dans [Cau02], Caucal a ouvert la voie à ce résultat. Dans cet article, il définit une sous-hiérarchie de la hiérarchie des arbres. Cette sous-hiérarchie $(\mathrm{Term}_n)_{n\in\mathbb{N}}$ ne contient que des arbres déterministes correspondant à des termes infinis[2]. Au niveau 0, Term_0 correspond à la classe des termes finis. Pour $k \geq 1$, Term_{k+1} est la classe des termes isomorphes à un terme obtenu par l'application d'une application rationnelle inverse déterministe[3] suivie d'un dépliage à un terme de Term_k. L'auteur établit, pour tout $k \geq 1$, que les termes de Term_{k+1} sont les solutions des schémas récursifs de niveau k dits *sûrs* tels qu'ils sont définis dans [KNU02]. Dans [KNU02], il est montré que les solutions des schémas sûrs de niveau k sont les termes acceptés par les automates à pile déterministes sur COps_k. Pour une définition précise de la notion d'acceptation par un automate à pile sur COps_k, nous renvoyons le lecteur à [KNU02]. Dans notre formalisme, ce résultat admet la formulation suivante:
les solutions des schémas récursifs de niveau k sont les termes isomorphes à la τ-fermeture du dépliage du graphe enraciné d'un automate à pile déterministe et normalisé sur COps_k.

2. Nous renvoyons le lecteur à [Cau02] pour la définition précise du codage des termes infinis comme des arbres déterministes. Dans la suite, nous parlerons de termes infinis ou finis pour désigner des arbres déterministes correspondant au codage d'un terme fini ou infini.

3. Une application rationnelle inverse déterministe est définie par une restriction syntaxique qui garantit que cette opération préserve le déterminisme des graphes auxquels elle est appliquée. Pour une définition précise, nous renvoyons le lecteur à [Cau02]

Avant d'établir, dans le sous-paragraphe 5.2.2, l'équivalence annoncée entre les graphes des transitions des automates sur Ops_k et les graphes de Graph_k, nous présentons une caractérisation naturelle des graphes des configurations des automates à pile sur Ops_k dans le sous-paragraphe suivant.

5.2.1 Graphes des configurations

Pour caractériser, par transformations de graphes, les graphes des configurations des automates à pile sur Ops_k, nous introduisons une sous-hiérarchie naturelle où la substitution rationnelle inverse est remplacée par l'application finie inverse. Le point de départ est la classe $\mathrm{Tree}_0^f = \mathrm{Tree}_0$ des arbres finis. Pour tout $k \geq 0$, on a ensuite:

$$\mathrm{Graph}_k^f := \big\{G \mid G \approx h^{-1}(T), T \in \mathrm{Tree}_k^f \text{ et } h\text{application finie inv.}\big\},$$
$$\mathrm{Tree}_{k+1}^f := \big\{T \mid T \approx \mathrm{Unf}(G,r), G \in \mathrm{Graph}_k^f \text{ et } r \in V_G\big\}.$$

Pour établir l'inclusion de la classe Graph_k^f dans la classe CHPDS_k, il suffit d'établir les propriétés de fermeture énoncées par la proposition ci-dessous.

Proposition 5.2.1.

1. *Pour tout $k \geq 1$, pour tout graphe G, pour toute application finie inverse h, si $G \in \mathrm{CHPDS}_k$ alors $h^{-1}(G) \in \mathrm{CHPDS}_k$,*
2. *Pour tout $k \geq 1$, pour tout graphe G et pour tout sommet $s \in V_G$, si $G \in \mathrm{CHPDS}_k$ alors $\mathrm{Unf}(G,s) \in \mathrm{CHPDS}_{k+1}$.*

Démonstration. Soit G un graphe appartenant à CHPDS_k. Pour simplifier, nous supposons que τ n'étiquette pas G. Si τ est une étiquette de G alors elle est traitée comme les autres symboles de Λ_G.

Par définition de CHPDS_k et par la remarque 5.1.7, il existe un automate à pile $\mathcal{A} = (\Gamma,\Sigma,\tau,\{\, q_0 \,\},\Delta_{\mathcal{A}})$ sur Ops_k et un ensemble rationnel $S \in \mathrm{Rat}_k$ tels que le graphe des configurations, noté H, pour la restriction S soit isomorphe au graphe G.

Soit h une application finie inverse. Notons Ξ le support nécessaire fini de h.

Considérons maintenant le morphisme φ du monoïde libre $(\Sigma \cup \overline{\Sigma})^*$ dans le monoïde des parties du monoïde engendré par $\mathrm{Ops}_k \cup \{\, \mathrm{Test}_S \,\}$ défini, pour tout $a \in \Sigma$, par:

$$\begin{cases} \varphi(a) &= \{\mathrm{Test}_S \cdot \theta \cdot \mathrm{Test}_S \mid (q_0,\theta,a,q_0) \in \Delta_{\mathcal{A}}\}, \\ \varphi(\overline{a}) &= \{\mathrm{Test}_S \cdot \overline{\theta} \cdot \mathrm{Test}_S \mid (q_0,\theta,a,q_0) \in \Delta_{\mathcal{A}}\}. \end{cases}$$

Nous étendons de manière canonique φ de $(\Sigma \cup \overline{\Sigma})^*$ aux parties de $(\Sigma \cup \overline{\Sigma})^*$.

Enfin, considérons l'automate à pile $\mathcal{B} = (\Gamma,\Sigma,\tau,\{\, q_0 \,\},\Delta_{\mathcal{B}})$ sur $\mathrm{Ops}_k \cup \{\, \mathrm{Test}_S \,\}$ dont l'ensemble des transitions est donné par:

$$\Delta_{\mathcal{B}} = \{(q_0,\theta',a,q_0) \mid \theta' \in \varphi(h(a)) \text{ et } a \in \Xi\}.$$

Notons K le graphe des configurations $\mathcal{B}$ pour la restriction à S. Il est aisé de vérifier que pour toutes piles s et $s' \in \mathrm{Stacks}_k(\Gamma)$ et pour tout $a \in \Xi$,

$$s \xrightarrow[H]{h(a)} s' \quad \Leftrightarrow \quad s \xrightarrow[K]{a} s'.$$

Il découle donc que K est égal à $h^{-1}(H)$. Par la proposition 5.1.11, le graphe K appartient à CHPDS_k: ce qui établit la fermeture de CHPDS_k par application finie inverse.

Soit $s \in S$. Nous montrons que $\mathrm{Unf}(H,s)$ appartient à CHPDS_{k+1}. Considérons l'automate à pile $\mathcal{B} = (\Gamma,\Sigma,\tau,\{\, q_0 \,\},\Delta_{\mathcal{B}})$ sur $\mathrm{Ops}_{k+1} \cup \{\, \mathrm{Test}_S \,\}$ dont l'ensemble des transitions est donné par:

$$\Delta_{\mathcal{B}} = \{(q_0,\mathrm{copy}_k \cdot \theta \cdot \mathrm{Test}_S,a,q_0) \mid (q_0,\theta,a,q_0) \in \Delta_{\mathcal{A}}\}.$$

Il est aisé de vérifier que le graphe $\mathcal{B}$ enraciné en $[\, s \,]_{k+1}$ est isomorphe au dépliage de $\mathrm{Unf}(H,s)$. Par les propositions 5.1.4 et 5.1.8, il découle donc que $\mathrm{Unf}(H,s)$ appartient à CHPDS_{k+1}. □

Par une récurrence immédiate sur le niveau k utilisant la propriété précédente, nous obtenons l'inclusion directe.

Proposition 5.2.2. *Pour tout $k \geq 1$, $\mathrm{Graph}_k^f \subseteq \mathrm{CHPDS}_k$.*

Considérons maintenant l'inclusion réciproque: il nous faut montrer que les graphes des configurations des automates à piles sur Ops_k appartiennent à la classe Graph_k^f. Par la remarque 5.1.7, nous pouvons supposer que l'alphabet de pile Γ est réduit à deux symboles. Pour le reste de ce paragraphe, nous fixons $\Gamma = \{\, a,b \,\}$.

L'étape clé dans l'obtention de ce résultat est de montrer que pour tout ensemble fini $\mathcal{L} = \{\, L_1,\ldots,L_n \,\} \subseteq \mathrm{Rat}_k$, l'arbre $\mathrm{TStacks}_k$ associé aux piles de niveau k sur Γ (cf. paragraphe 4.7) «marqué» par les langages de $\mathcal{L}$ appartient à Tree_k^f. Formellement, le marquage de $\mathrm{TStacks}_k$ par $\mathcal{L}$ est un arbre noté $[\![\, \mathrm{TStacks}_k \,]\!]_{\mathcal{L}}$ et étiqueté par $\Gamma_k^{\circ} \cup \{\, \$_1,\ldots,\$_n \,\}$ qui est égal à:

$$\mathrm{TStacks}_k \cup \{(s,\$_i,(i,s)) \mid s \in \mathrm{Stacks}_k, i \in [1,n] \text{ et } s \in L_i\}.$$

Cette définition est illustrée dans la figure 5.6.

Proposition 5.2.3. *Pour tout $k \geq 1$ et pour tout ensemble fini $\mathcal{L} \subseteq Rat_k(\Gamma)$, l'arbre $[\![\, \mathrm{TStacks}_k \,]\!]_{\mathcal{L}}$ appartient à Tree_k^f.*

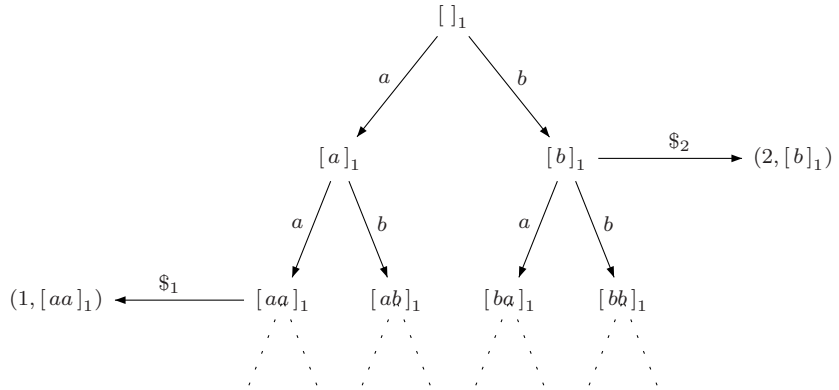


FIG. 5.6 – *L'arbre* $[\![\,\mathrm{TStacks}_1\,]\!]_{\mathcal{L}}$ *pour* $\mathcal{L} = \{\, L_1 = aa^+, L_2 = b(bb)^* \,\}$.

Démonstration. Avant de passer à la preuve de cette propriété, remarquons que pour tout $k \geq 1$ les classes Tree_k^f et Graph_k^f sont fermées par l'opération de copie par un ensemble fini (cf. paragraphe 3.2).

La preuve procède par récurrence sur le niveau k. Au niveau 1, il est évident que $[\![\,\mathrm{TStacks}_1\,]\!]_{\mathcal{L}}$ est un arbre régulier et donc appartient à $\mathrm{Tree}_1^f = \mathrm{Tree}_1$. Supposons la propriété établie au niveau k et montrons qu'elle est vraie au niveau $k+1$.

Soit $\mathcal{L} = \{\, L_1, \ldots, L_n \,\} \subseteq \mathrm{Rat}_{k+1}$. Il existe un automate $A = (Q, \{\, i \,\}, F, \Delta)$ déterministe et complet sur Γ_{k+1} avec tests dans Rat_k et une relation $\eta \subseteq Q \times [1,n]$ tels que pour toute pile $s \in \mathrm{Stacks}_{k+1}(\Gamma)$ et pour tout $i \in [1,n]$,

$$s \in L_i \Leftrightarrow (A(s), i) \in \eta.$$

Nous renvoyons le lecteur à la preuve de la proposition 5.1.4 pour l'existence de cet automate. Notons $\mathcal{L}' = \{\, L'_1, \ldots, L'_m \,\} \subseteq \mathrm{Rat}_k$ l'ensemble des langages de tests apparaissant dans A.

Par hypothèse de récurrence, le graphe $T = [\![\,\mathrm{TStacks}_k\,]\!]_{\mathcal{L}'}$ appartient à Tree_k^f. Le graphe G obtenu rajoutant à T pour tout arc (s, x, s') avec $x \in \Gamma_k^{\mathrm{o}}$, un arc $(s', \overline{x}, s)$ et rajoutant une boucle étiqueté par k sur tous les sommets $s \in \mathrm{Stacks}_k$ appartient à Graph_k^f. La construction que nous présentons est intuitivement un produit synchronisé entre le graphe G et l'automate fini A. La réalisation de ce produit synchronisé à l'aide de la copie par un ensemble fini et de la substitution finie inverse est adapté de [Urv03].

Notons H le graphe obtenu en appliquant la copie par l'ensemble fini Q puis par l'ensemble $\{\, \sharp_1, \ldots, \sharp_n \,\}$ au graphe G. Comme nous l'avons remarqué au début de cette preuve, G appartient à Graph_k^f.

Enfin considérons la substitution finie[4] inverse h définie par:

$$\begin{cases} h(x) = \{\underline{p}\, x\, \underline{\$_{i_1}}\, \$_{i_1} \ \ldots\ \underline{\$_{i_\ell}}\, \$_{i_\ell}\, q \mid p \xrightarrow{x} q, \{ T_{L_{i_1}}, \ldots, T_{L_{i_\ell}} \} \in \Delta \} \\ h(\$_\ell) = \{\underline{\sharp_\ell}\, \underline{p}\, p\, \sharp_\ell \mid p \in \overline{Q} \text{ et } (q,\ell) \in \eta \} \end{cases}$$

pour $x \in \Gamma_k^{\circ} \cup \{ k \}$ et pour tout $\ell \in [1,n]$.

On vérifie que le dépliage de $h^{-1}(G)$ à partir du sommets $(i, [\ ]_k)$ est isomorphe au graphe $[\![\text{TStacks}_{k+1}]\!]_{\mathcal{L}}$. □

Nous pouvons maintenant établir l'inclusion réciproque.

Proposition 5.2.4. *Pour tout $k \geq 1$,* $\text{Graph}_k^f \supseteq \text{CHPDS}_k$.

Démonstration. Soit G un graphe de CHPDS_k. Par définition de CHPDS_k et par la remarque 5.1.7, il existe un automate à pile $\mathcal{A} = (\Gamma,\Sigma,\tau,\{ q_0 \},\Delta_{\mathcal{A}})$ et un ensemble rationnel R de piles de niveau k tels que G soit isomorphe au graphe H des configurations de $\mathcal{A}$ pour la restriction R. Par la proposition 5.2.3, l'arbre $[\![\text{TStacks}_k]\!]_{\mathcal{L}}$ avec $\mathcal{L} = \{ R \}$ appartient à Tree_k^f. Le graphe K obtenu rajoutant à $[\![\text{TStacks}_k]\!]_{\mathcal{L}}$ pour tout arc (s,x,s') avec $x \in \Gamma_k^{\circ}$, un arc $(s',\overline{x},s)$ appartient donc à Graph_k.

Considérons la substitution finie inverse h définie pour tout $x \in \Sigma \cup \{ \tau \}$ par:

$$h(x) = \{\$_1 \overline{\$_1} \mathcal{R}^{-1}(\theta) \$_1 \overline{\$_1} \mid (p_0,\theta,x,p_0\}.$$

Il est aisé vérifier que $h^{-1}(K)$ est isomorphe au graphe H. Il découle donc que G appartient à Graph_k^f. □

En combinant les propositions 5.2.2 et 5.2.4, nous dérivons la caractérisation externe suivante des graphes des configurations des automates sur Ops_k.

Théorème 5.2.5. *Pour tout $k \geq 1$,* $\text{Graph}_k^f = \text{CHPDS}_k$.

5.2.2 Graphes des transitions

Dans ce sous-paragraphe, nous établissons l'égalité entre les classes Graph_k et GHPDS_k. Pour établir l'inclusion de la classe Graph_k dans la classe GHPDS_k, il suffit d'établir les propriétés de fermeture énoncées par la proposition ci-dessous.

Proposition 5.2.6.

1. *Pour tout $k \geq 1$, pour tout graphe G, pour toute substitution rationnelle inverse h^{-1}, si $G \in \text{GHPDS}_k$ alors $h^{-1}(G) \in \text{GHPDS}_k$,*

4. Pour supprimer la confusion entre la notation barrée pour les instructions de Γ_k° et la notation barrée utilisée dans la définition des substitutions finies inverse, nous utiliserons la notation $\underline{a}$ au lieu de la notation $\overline{a}$ dans la définition des substitutions finies inverses.

2. *Pour tout* $k \geq 1$, *pour tout graphe* G *et pour tout sommet* $s \in V_G$, *si* $G \in \mathrm{CHPDS}_k$ *alors* $\mathrm{Unf}(G,s) \in \mathrm{GHPDS}_{k+1}$.

Démonstration. Soit G un graphe appartenant à GHPDS_k. Par le théorème 5.1.20, G est isomorphe à un graphe préfixe-reconnaissable H défini par $(P_a)_{a\in\Sigma}$. Comme les relations prefixe-reconnaissables sont fermées par inverse, concaténation, union et fermeture transitive (cf. théroème 4.5.16), les graphes préfixe-reconnaissables sont fermés par substitution rationnelle inverse.

Soit $s \in V_H$. Montrons que $\mathrm{Unf}(H,s)$ est un graphe de GHPDS_{k+1}. Par la proposition 4.5.13, V_H est un ensemble rationnel de piles. Considérons l'ensemble rationnel R de piles de niveau $k+1$ égal à $\{[\, s_1,\ldots,s_n\,]_{k+1} \mid \forall i \in [1,n],(s_i,s_{i+1}) \in \cup_{a\in\Sigma} P_a \text{ et } s_1 = s\}$ et le graphe préfixe-reconnaissable K défini par la famille $(Q_a)_{a\in\Sigma}$ où $Q_a = \mathrm{Test}_R \cdot \mathrm{copy}_k \cdot P_a \cdot \mathrm{Test}_R$. Il est aisé de vérifier que K est isomorphe à $\mathrm{Unf}(H,s)$ et donc $\mathrm{Unf}(H,s)$ appartient à GHPDS_{k+1}. □

L'inclusion réciproque est une conséquence assez immédiate de la proposition 5.2.4.

Proposition 5.2.7. *Pour tout* $k \geq 1$, $\mathrm{Graph}_k^f \supseteq \mathrm{GHPDS}_k$.

Démonstration. Soit G un graphe de GHPDS_k. Par définition de GHPDS_k et par la remarque 5.1.7, il existe un automate à pile $\mathcal{A} = (\Gamma,\Sigma,\tau,\{\, q_0\,\},\Delta_{\mathcal{A}})$ et un ensemble rationnel R de piles de niveau k tels que G soit isomorphe à la τ-fermeture du graphe H des configurations de $\mathcal{A}$ pour la restriction R. D'après la proposition 5.1.14, l'ensemble O des configurations observables de H est un ensemble rationnel. Considérons l'automate à pile $\mathcal{B}$ obtenu en rajoutant une transition $(q_0,\mathrm{Test}_O,\$,q_0)$ aux transitions de l'automate $\mathcal{A}$ pour \$ un symbole n'appartenant pas à Σ. Le graphe des configurations, noté K, de $\mathcal{B}$ pour la restriction R est égal au graphe H auxquel une boucle étiquetée par \$ est ajoutée sur chaque configuration observable. Par la proposition 5.1.11, le graphe K appartient à CHPDS_k. Il est aisé de vérifier que la substitution rationnelle inverse h définie par $h(a) = \$a\tau^*\$$ pour tout $a \in \Sigma$ est telle que $h^{-1}(K)$ est égal à la τ-fermeture de H. Il découle donc que G appartient à GHPDS_k. □

Théorème 5.2.8. *Pour tout* $k \geq 1$, $\mathrm{Graph}_k = \mathrm{GHPDS}_k$.

Nous allons maintenant établir quelques propriétés de fermeture des classes GHPDS_k.

Théorème 5.2.9. *La classe* GHPDS *est fermée par interprétation monadique.*

Démonstration. Par la proposition 5.2.3, l'arbre $\mathrm{TStacks}_k$ appartient à Tree_k. Par les théorèmes 4.7.5 et 5.1.20, tout graphe G de GHPDS_k est isomorphe à un

graphe interprétable dans $\mathrm{TStacks}_k$. Il existe donc une interprétation monadique $\mathcal{J}_G$ telle que $G \approx \mathcal{J}_G(\mathrm{TStacks}_k)$.

Soit $\mathcal{I}$ une interprétation monadique et G un graphe GHPDS_k. Le graphe $\mathcal{I}(G)$ est interprétable dans $\mathrm{TStacks}_k$ (*i.e.* $\mathcal{I}(G) \approx \mathcal{J}_G \circ \mathcal{I}(\mathrm{TStacks}_k)$). D'après le théorème 4.7.5, $\mathcal{I}(G)$ est isomorphe à un graphe préfixe-reconnaissable de niveau k. □

Nous dérivons des résultats présentés dans le chapitre 3 les corollaires suivants.

Corollaire 5.2.10. *La classe* GHPDS_k *est fermée par transduction monadique et interprétation monadique avec quotient.*

Démonstration. Il est aisé de vérifier que la classe GHPDS_k est fermée par l'opération de copie par un ensemble fini. De plus comme elle est fermée par interprétation, elle est close par transduction monadique. Soit G un graphe appartenant à GHPDS_k et $\mathcal{I}_\varepsilon$ une interprétation monadique avec quotient. Par un raisonnement similaire à celui de la preuve du théorème 5.2.9, on établit l'existence d'une interprétation monadique avec quotient $\mathcal{J}_\varepsilon$ telle que $\mathcal{I}_\varepsilon(G)$ soit isomorphe à $\mathcal{J}_\varepsilon(\mathrm{TStacks}_k)$. Par la propostion 3.1.4, il existe une transduction monadique $\mathcal{T}$ telle que $\mathcal{J}_\varepsilon(\mathrm{TStacks}_k) \approx \mathcal{T}(\mathrm{TStacks}_k)$. Donc comme GHPDS_k est fermée par transduction, le graphe $\mathcal{I}_\varepsilon(G) \approx \mathcal{J}_\varepsilon(\mathrm{TStacks}_k) \approx \mathcal{T}(\mathrm{TStacks}_k)$ appartient à GHPDS_k. □

Corollaire 5.2.11. *Pour tout* $G \in \mathrm{GHPDS}_k$ *et pour tout* $\sharp \notin \Lambda_G$*, le graphe* $\mathrm{Treegraph}(G,\sharp)$ *appartient à* GHPDS_{k+1}.

Démonstration. Soit G un graphe de GHPDS_k et soit $\sharp \notin \Lambda_G$.

D'après la proposition 4.7.1, $\mathrm{TStacks}_k$ et $\mathrm{GStacks}_k$ sont mutuellement interprétables. Par le théorème 5.2.9 et comme $\mathrm{TStacks}_k$ appartient Tree_k, nous avons que $\mathrm{GStacks}_k$ appartient à GHPDS_k.

Nous avons vu dans la preuve du théorème 5.2.9 qu'il existe une interprétation monadique $\mathcal{I}_G$ telle que $G \approx \mathcal{I}_G(\mathrm{TStacks}_k)$ et donc il existe une interprétation monadique $\mathcal{J}_G$ telle que $G \approx \mathcal{J}_G(\mathrm{GStacks}_k)$.

Par la proposition 3.4.1, il existe une interprétation monadique $\mathcal{J}_0$ telle que

$$\begin{aligned} \mathrm{Treegraph}(G,\sharp) &\approx \mathrm{Treegraph}(\mathcal{J}_G(\mathrm{GStacks}_k),\sharp) \\ &\approx \mathcal{J}_0(\mathrm{Treegraph}(\mathrm{GStacks}_k,k)) \\ &= \mathcal{J}_0(\mathrm{GStacks}_{k+1}). \end{aligned}$$

□

Une conséquence de ces deux corollaires est que si nous remplaçons le dépliage par l'opération de Treegraph et la substitution rationnelle inverse par la transduction monadique nous n'augmentons pas la hiérarchie.

5.3 Générateur et propriétés logiques

Pour tout $k \geq 1$, tout graphe $G \in \mathrm{Graph}_k$ possède une théorie monadique décidable. Cette propriété ne découle pas immédiatement de la définition de cette hiérarchie car le dépliage peut s'effectuer à partir de n'importe quel sommet et non à partir d'un sommet MSO-définissable (cf. remarque 3.3.5). Cependant on ne modifie pas la hiérarchie obtenue en imposant que le dépliage ne se fasse qu'à partir d'un sommet MSO-definissable. Comme le dépliage à partir d'un sommet MSO-definissable et la substitution rationnelle inverse sont des transformations de graphes MSO-compatibles, il suit alors que tous les graphes de Graph_k ont une théorie monadique décidable. La complexité de cette procédure de décision est non-élémentaire. Dans [Cac03b], Cachat établit que la décision du μ-calcul pour un graphe du niveau k peut être effectuée en temps $\exp[k]$. Au niveau 1, ce résultat a été obtenu par Walukiewicz dans [Wal96b].

Comme nous l'avons vu dans le paragraphe précédent, les deux graphes $\mathrm{TStacks}_k$ et $\mathrm{GStacks}_k$ associées aux piles de niveau k jouent un rôle privilégié. En effet, ces deux graphes sont des générateurs pour l'interprétation monadique.

Théorème 5.3.1. *Pour tout $k \geq 1$,*

$$\begin{aligned} \mathrm{GHPDS}_k &= \{G \mid G \approx \mathcal{I}(\mathrm{GStacks}_k) \text{ et } \mathcal{I} \text{ int. monadique}\} \\ &= \{G \mid G \approx \mathcal{I}(\mathrm{TStacks}_k) \text{ et } \mathcal{I} \text{ int. monadique}\} \end{aligned}$$

Remarque 5.3.2. Nous pouvons remplacer les interprétations monadiques par des transductions dans l'énoncé précédent.

Nous pouvons maintenant transférer les résultats obtenus sur les graphes $\mathrm{GStacks}_k$ et $\mathrm{TStacks}_k$ dans le paragraphe 4.7. Comme nous allons le voir, c'est propriété ne sont plus «à isomorphisme près» mais dépendent du nommage des sommets.

Théorème 5.3.3. *Pour tout automate à pile $\mathcal{A}$ sur Ops_k, les ensembles définissables en logique monadique sur un graphe des configurations (resp. un graphe des transitions) de $\mathcal{A}$ sont des ensembles rationnels de configurations. Si un graphe G des configurations de $\mathcal{A}$ satisfait une formule $\exists X, \varphi(X)$ alors il existe un ensemble rationnel de configurations R tel que $G \models \varphi[R]$.*

Remarque 5.3.4. Les graphes des configurations et des transitions ne satisfont pas nécessairement la propriété de sélection. En effet l'ensemble R de configurations n'est pas nécessaire définissable en logique monadique.

Un corollaire très interessant de ce résultat permet d'établir entre le déterminisme des graphes des transitions et le déterminisme du graphe des configurations sous-jacent. En effet comme nous l'avons vu dans le paragraphe 2.1.2 le graphe

des transitions peut-être déterministe sans que le graphes des configurations dont il est la τ-fermeture le soit.

Corollaire 5.3.5. *Un graphe déterministe de* GHPDS_k *est isormorphe à la* τ*-fermeture d'un graphe déterministe de* CHPDS_k*.*

Nous concluons ce paragraphe en récapitulant les différentes caractérisations des graphes de GHPDS_k que nous avons obtenue. À part la caractérisation par graphes préfixe-reconnaissables de niveau k, ces résultats ont été obtenue en collaboration avec Stefan Wöhrle et une version préliminaire. Dans cette version, cependant nous n'utilisons pas la rationalité pour les piles de piles et nous n'avons donc pas de notion de graphes des configurations et les preuves s'en trouvent complexifiées.

Théorème 5.3.6. *Les propositions suivantes sont équivalentes à isomorphisme près:*

1. *G est un graphe des transitions d'un automates à piles sur* Ops_k*,*
2. *G est un graphe des transitions d'un automates à piles sur* COps_k*,*
3. *G est un graphe préfixe-reconnaissable de niveau k,*
4. *G est un obtenu en itérant k fois l'application d'une substitution rationnelle inverse suivi d'un dépliage en partant d'un graphe fini,*
5. *G est un obtenu en itérant k fois l'application d'une transduction monadique suivie d'une opération de* Treegraph *en partant d'un graphe fini,*
6. *G est interprétable dans* $\mathrm{GStacks}_k$ *(resp.* $\mathrm{TStacks}_k$*).*

5.4 Traces

Une conséquence immédiate du théorème 5.1.23 est que les traces des graphes des transitions des automates entre un unique sommet initial et un unique sommet final sont les langages indexés de niveau k.

Proposition 5.4.1. *Pour tout niveau $k \geq 1$, les traces des graphes de* GHPDS_k *entre deux sommets sont les langages indexés de niveau k.*

Remarquons que les traces des graphes des transitions des automates à pile sur Ops_k entre deux ensembles rationnels de configurations sont elles aussi des langages indexés de niveau k.

Comme la hiérarchie des langages indexés est stricte (cf. théorème 4.1.16), nous pouvons déduire de la proposition précédente que la hiérarchie $(\mathrm{GHPDS}_k)_{k\geq 1}$ des graphes des transitions est elle aussi stricte.

Théorème 5.4.2. *Pour tout $k \geq 1$,* $\mathrm{GHPDS}_k \subsetneq \mathrm{GHPDS}_{k+1}$.

Le générateur GStacks_{k+1} est un exemple de graphe appartenant à GHPDS_{k+1} mais pas à GStacks_k. Au vu des nombreuses propriété de clotûre de cette hiérarchie, il est naturel de chercher à donner un graphe de théorie monadique décidable n'appartenant pas à cette hiérarchie.

Prenons $\exp_\omega(0) := 0$ et $\exp_\omega(n) := 2^{\exp_\omega(n-1))}$. L'arbre $T_{\exp_\omega}$ associé à la fonction $\exp_\omega$ est présenté à la figure 5.7. Il est constitué d'un demie-droite infini de sommets reliés entre eux par des arcs étiquetés par a. Du sommet $n^{\text{ième}}$ sommet de cette demi-droite par un segment étiqueté par b de longueur $\exp_{\omega(n)}$.

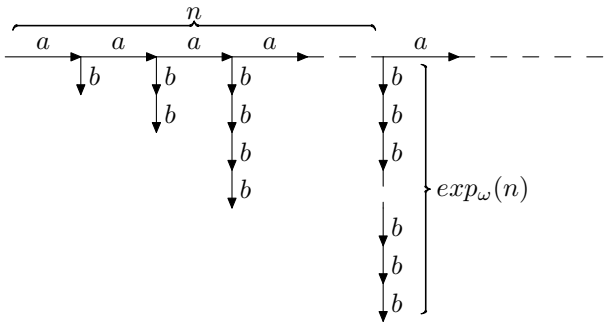


FIG. 5.7 – *L'arbre* T_{exp_ω}.

Proposition 5.4.3. *L'arbre* $T_{\exp_\omega}$ *a une théorie monadique décidable mais n'appartient pas à* $\bigcup_{k\in\mathbb{N}} \text{GHPDS}(k)$.

Démonstration. La décidabilité de la théorie monadique $T_{\exp_\omega}$ est établie par exemple dans [MP04]. Pour montrer $T_{\exp_\omega}$ n'est pas un graphe de transition d'automate à pile, il suffit de montrer que le langage $L_{\exp_\omega} = \{a^n b^{\exp_\omega(n)} \mid n \geq 1\}$ n'est pas un langage indexé d'ordre supérieur. En effet si $T_{\exp_\omega}$ appartient à $\bigcup_{k\in\mathbb{N}} \text{GHPDS}(k)$ alors sont langage des branches $L_{\exp_\omega}$ est une langage indexé d'ordre supérieur.

Pour cela nous utilisons une propriété des indexes rationnels des les langages indexés de niveau k établit dans [Dam82].

L'index rationnel d'un langage L est une fonction f_L de $\mathbb{N}$ dans $\mathbb{N}$ associant à tout $n \geq 1$ le $\max\{d(L,R) \mid R \in \text{Rat}_n \text{ et } R \cap L \neq \emptyset\}$ où Rat_n est l'ensemble des langages acceptés par un automate fini ayany au plus n états et où $d(L,R)$ désigne la longueur du plus petit élément de $L \cap R$.
Dans [Dam82], il est établit que si un langage L est indexé de niveau k alors f_L est bornée par une fonction de $\exp[2k]$.
Cependant pour tout $n \in \mathbb{N}$, $f_{L_{\exp_\omega}}(n+1) \geq \exp_\omega(n)$. En effet, $d(a^n b^*, L_{\exp_\omega}) = \exp_\omega(n)$ et $a^n b^* \in \text{Rat}_{n+1}$. Donc $L_{\exp_\omega}$ n'est un langage indexé d'ordre supérieur et donc $T_{\exp_\omega}$ n'appartient pas à $\bigcup_{k\in\mathbb{N}} \text{GHPDS}(k)$. □

Perspectives

Pour conclure, nous présentons quelques problèmes ouverts et des perspectives liés aux travaux présentés dans ce document.

Autour des traces

Une question importante dans le cadre de notre étude est la place des langages indexés d'ordre supérieur dans la hiérarchie de Chomsky. Dans [Aho69], Aho établit que les langages indexés (*i.e.* acceptés par des automates à pile de niveau 2) sont des langages contextuels. Nous conjecturons que les langages indexés d'ordre supérieur sont des langages contextuels déterministes. Nous pensons que la notion de graphe préfixe-reconnaissable de niveau k permettra d'établir ce résultat de manière uniforme.

Si cette conjecture est vérifiée, nous pourrons établir que les graphes des transitions des automates à pile sur Ops_k de degré sortant borné sont des graphes linéairement bornés (cf. paragraphe 2.3.2). La preuve de ce résultat suivrait un raisonnement similaire à celui développé, dans [CM05], pour établir que les graphes rationnels de degré sortant borné sont des graphes linéairement bornés.

Ceci nous amène à considérer un autre problème: les graphes de transitions des automates à pile sur Ops_k de degré sortant borné tracent-ils tous les langages indexés de niveau k?

Rappelons que nous conjecturons que contrairement au niveau 1 les graphes de transitions de niveau k de degré (entrant et sortant) borné contiennent strictement les graphes de configurations de niveau k. Du point de vue de la théorie des langages formels, cette conjecture peut se reformuler de la manière suivante: les automates sur Ops_k temps réel sont strictement moins expressifs que les automates sur Ops_k.

Autour de la rationnalité sur les piles de piles

Nous avons vu que les ensembles rationnels de piles de niveau k sont les ensembles définissables en logique monadique sur $\mathrm{GStacks}_k$. Cependant la com-

plexité de la transformation d'une formule $\varphi(x)$ en un automate fini sur Γ_k acceptant le langage de piles de niveau k défini par φ est non-élémentaire. Nous pensons qu'une adaptation des techniques présentées dans ce document permettra d'obtenir une complexité élémentaire pour la construction d'automates finis sur Γ_k acceptant les ensembles définis par des formules du μ-calcul sur $\mathrm{GStacks}_k$ étendu avec toutes les instructions de Γ_k. Plus généralement, il serait intéressant d'étudier les applications des résultats présentés dans ce document au *regular model-checking* des automates à pile d'ordre supérieur (voir par exemple [BEM97, BM04]).

Dans [KNUW05], les auteurs introduisent la notion d'automate avec panique qui est un enrichissement de la notion d'automate à pile de niveau 2. Ces automates acceptent les solutions des schémas récursifs non-sûrs. Il serait intéressant d'étudier la notion d'ensemble rationnel de piles associé à ces automates.

En collaboration avec Didier Caucal [CC06], nous avons proposé un cadre permettant d'expliquer des résultats de fermeture par complémentaire des ensembles rationnels d'un monoïde par une notion d'acceptation déterministe par automate fini. Il serait intéressant d'étendre ce cadre pour pouvoir capturer les ensembles rationnels de piles de niveau k.

Autour des graphes des automates à pile de piles

L'étude de la hiérarchie des graphes des transitions des automates à pile d'ordre supérieur laisse de nombreuses questions ouvertes. En particulier, nous disposons de très peu d'outils pour prouver qu'un graphe n'appartient pas à un niveau donné de cette hiérarchie. Ce problème apparaît déjà pour les langages indexés où contrairement aux langages algébriques on ne dispose pas d'outils pour montrer qu'un langage n'est pas indexé de niveau k. Dès le niveau 2, les lemmes de pompage deviennent impraticables (voir par exemple [Hay73]).

Au niveau 1, les graphes des configurations admettent une caractérisation géométrique très élégante dûe à Muller et Schupp. Il est naturel de se demander si cette approche peut être étendue aux niveaux supérieurs.

Enfin la question qui nous semble la plus intéressante est comment obtenir une famille d'automates infinis ayant une théorie au second ordre monadique décidable qui étendent la hiérarchie des automates à pile d'ordre supérieur. Récemment, il a été montré que les solutions des schémas récursifs non sûrs de tous niveaux ont une théorie MSO décidable [KNUW05, AdMO05, Ong06]. Les auteurs conjecturent que les solutions de ces schémas n'appartiennent pas à la hiérarchie des automates à pile d'ordre supérieur.

L'étude structurelle des automates infinis se révèle d'une grande richesse. Bien que cette thématique en soit à ses débuts, on peut espérer qu'elle atteigne un jour

le niveau de compréhension et de finesse de la théorie des automates finis.

Bibliographie

[AdMO05] K. Aehlig, J. G. de Miranda, and C.-H. Luke Ong. The monadic second order theory of trees given by arbitrary level-two recursion schemes is decidable. In *Proceedings of the 7th International Conference on Typed Lambda Calculi and Applications (TACAS 2005)*, volume 3461 of *Lecture Notes in Computer Science*, pages 39–54. Springer, 2005.

[Aho68] A. Aho. Indexed grammars, an extension of context-free grammars. *Journal of the Association for Computing Machinery (ACM)*, 15:647–671, 1968.

[Aho69] A. Aho. Nested stack automata. *Journal of the Association for Computing Machinery (ACM)*, 16:383–406, 1969.

[Bar97] K. Barthelmann. On equational simple graphs. Technical Report 9, Universität Mainz, Institut für Informatik, 1997.

[BEM97] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model-checking. In *Proceedings of the 8th International Conference on Concurrency Theory (CONCUR 1997)*, volume 1243 of *Lecture Notes in Computer Science*, pages 135–150. Springer, 1997.

[Ben69] M. Benois. Parties rationnelles du groupe libre. *Comptes-Rendus de l'Acamdémie des Sciences de Paris, Série A*, 269:1188–1190, 1969.

[Ber79] J. Berstel. *Transductions and Context-Free Languages*. Leitfäden der angewandten Mathematik und Mechanik. Teubner, 1979.

[BG00] A. Blumensath and E. Grädel. Automatic structures. In *Proceedings of the 15th IEEE Symposium on Logic in Computer Science (LICS 2000)*, pages 51–62. IEEE, 2000.

[BH67] M. Blum and C. Hewitt. Automata on a 2-dimensional tape. In *Proceedings of the 8th IEEE Symposium on Swithching and Automata Theory*, pages 155–160, 1967.

[Blu01] A. Blumensath. Prefix-recognizable graphs and monadic second order logic. Technical Report AIB-06-2001, RWTH Aachen, 2001.

[Blu03] A. Blumensath. *Structures of Bounded Partition Width.* PhD thesis, RWTH Aachen, 2003.

[BM04] A. Bouajjani and A. Meyer. Symbolic reachability analysis of higher-order context-free processes. In *Proceedings of the 24th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2004)*, volume 3328 of *Lecture Notes in Computer Science*, pages 135–147. Springer, 2004.

[BS86] M. Benois and J. Sakarovitch. On the complexity of some extended word problems defined by cancellation rules. *Information Processing Letters*, 23(6):281–287, 1986.

[Büc62] J. Büchi. On a decision method in restricted second order arithmetic. In *Proceedings of the 1960 International Congress of Logic, Methodology and Philosophy of Science*, pages 1–12. Stanford University Press, 1962.

[Büc64] J. Büchi. Regular canonical systems. *Arch. Math. Logik Grundlag.*, 6:91–111, 1964.

[Cac03a] T. Cachat. *Games on pushdown graphs and extensions.* PhD thesis, RWTH Aachen, 2003.

[Cac03b] T. Cachat. Higher order pushdown automata, the Caucal hierarchy of graphs and parity games. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP 2003)*, volume 2719 of *Lecture Notes in Computer Science*, pages 556–569. Springer, 2003.

[Car05] A. Carayol. Regular sets of higher-order pushdown stacks. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science (MFCS 2005)*, volume 3618 of *Lecture Notes in Computer Science*, pages 168–179. Springer, 2005.

[Cau92] D. Caucal. On the regular structure of prefix rewriting. *Theoretical Computer Science*, 106:61–86, 1992.

[Cau95] D. Caucal. Bisimulation of context-free grammars and of pushdown automata. In A. Ponse, M. de Rijke, and Y. Venema, editors, *Modal Logic and Process Algebra*, volume 53 of *CSLI Lecture Notes*, pages 85–106. Stanford, 1995.

[Cau96] D. Caucal. On infinite transition graphs having a decidable monadic theory. In *Proceedings of the 23rd International Colloquium on Automata, Languages, and Programming (ICALP 1996)*, volume 1099 of *Lecture Notes in Computer Science*, pages 194–205. Springer, 1996.

[Cau98] D. Caucal. *Sur des graphes infinis réguliers.* Habilitation thesis, Université de Rennes 1, 1998.

[Cau02] D. Caucal. On infinite terms having a decidable monadic theory. In *Proceedings of the 27th International Symposium on Mathematical Foundations of Computer Science (MFCS 2002)*, volume 2420 of *Lecture Notes in Computer Science*, pages 165–176. Springer, 2002.

[Cau03a] D. Caucal. On infinite transition graphs having a decidable monadic theory. *Theoretical Computer Science*, 290:79–115, 2003.

[Cau03b] D. Caucal. On the transition graphs of Turing machines. *Theoretical Computer Science*, 296:195–223, 2003.

[CC03] A. Carayol and T. Colcombet. On equivalent representations of infinite structures. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP 2003)*, volume 2719 of *Lecture Notes in Computer Science*, pages 599–610. Springer, 2003.

[CC06] A. Carayol and D. Caucal. The Kleene equality for graphs. In *Proceedings of the 31th International Symposium on Mathematical Foundations of Computer Science (MFCS 2006)*, volume 4162 of *Lecture Notes in Computer Science*, pages 214–225. Springer, 2006.

[CDG+] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: http://www.grappa.univ-lille3.fr/tata.

[Cho59] N. Chomsky. On certain formal properties of grammars. *Information and Control*, 2:137–167, 1959.

[CK01] D. Caucal and T. Knapik. An internal presentation of regular graphs by prefix-recognizable graphs. *Theoretical Computer Science*, 34:299–336, 2001.

[CK02] D. Caucal and T. Knapik. A Chomsky-like hierarchy of infinite graphs. In *Proceedings of the 27th International Symposium on Mathematical Foundations of Computer Science (MFCS 2002)*, volume 2420 of *Lecture Notes in Computer Science*, pages 177–187. Springer, 2002.

[CKS81] A. K. Chandra, D. Kozen, and L. Stockmeyer. Alternation. *Journal of the Association for Computing Machinery (ACM)*, 28(1):114–133, 1981.

[CL06] T. Colcombet and C. Löding. Transforming structures by set interpretations. Technical Report AIB-2006-07, RWTH Aachen, May 2006.

[CM05] A. Carayol and A. Meyer. Linearly bounded infinite graphs. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science (MFCS 2005)*, volume 3618 of *Lecture Notes in Computer Science*, pages 180–191. Springer, 2005.

[CM06a] A. Carayol and A. Meyer. Context-sensitive languages, rational graphs and determinism. *Logical Methods in Computer Science*, 2006.

[CM06b] A. Carayol and A. Meyer. Linearly bounded infinite graphs (long version). To appear in Acta Informatica, 2006.

[CM06c] A. Carayol and C. Morvan. On rational trees. In *Proceedings of the Annual Conference of the European Association for Computer Science Logic (CSL 2006)*, volume 4207 of *Lecture Notes in Computer Science*, pages 225–239. Springer, 2006.

[Col04] T. Colcombet. *Représentations et propriétés de structures infinies*. PhD thesis, Université de Rennes 1, France, 2004.

[Cou89] B. Courcelle. The monadic second-order logic of graphs, II: Infinite graphs of bounded width. *Mathematical System Theory*, 21:187–221, 1989.

[Cou90] B. Courcelle. Graph rewriting: An algebraic and logic approach. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics*, pages 193–242. Elsevier, 1990.

[Cou94] B. Courcelle. Monadic second-order definable graph transductions: A survey. *Theoretical Computer Science*, 126(1):53–75, 1994.

[Cou97] B. Courcelle. The expression of graph properties and graph transformations in monadic second-order logic. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation. Vol. I: Foundations*, chapter 5, pages 313–400. World Scientific, 1997.

[Cou03] B. Courcelle. The monadic second-order logic of graphs XIV: Uniformly sparse graphs and edge set quantifications. *Theoretical Computer Science*, 299:1–36, 2003.

[CW98] B. Courcelle and I. Walukiewicz. Monadic second-order logic, graph coverings and unfoldings of transition systems. *Annals of Pure and Applied Logic*, 92:35–62, 1998.

[CW03] A. Carayol and S. Wöhrle. The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In *Proceedings of the 23rd International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2003)*, volume 2914 of *Lecture Notes in Computer Science*, pages 112–123. Springer, 2003.

[Dam82] W. Damm. The IO- and OI-hierarchies. *Theoretical Computer Science*, 20:95–207, 1982.

[DG86] W. Damm and A. Goerdt. An automata-theoretical characterization of the. *Information and Control*, 71:1–32, 1986.

[EF95] H.D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer, 1995.

[EH99] J. Engelfriet and H. J. Hoogeboom. Tree-walking pebble automata. In *Jewels are Forever, Contributions on Theoretical Computer Science in Honor of Arto Salomaa*, pages 72–83. Springer, 1999.

[EJ91] E.A. Emerson and C.S. Jutla. Tree automata, μ-calculus and determinacy. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science (FoCS 1991)*, pages 368–377, 1991.

[EM65] C. Elgot and J. Mezei. On relations defined by finite automata. *IBM Journal of Research and Development*, 9:47–68, 1965.

[Eng83] J. Engelfriet. Iterated pushdown automata and complexity classes. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC 1983)*, pages 365–373. ACM Press, 1983.

[Eng91] J. Engelfriet. Iterated stack automata and complexity classes. *Information and Computation*, 95(1):21–75, 1991.

[For00] L. Fortnow. Diagonalization. *Bulletin of the European Association for Theoretical Computer Science*, 71:102–112, 2000.

[Fra05] S. Fratani. *Automates à pile de piles ... de piles*. PhD thesis, Université de Bordeaux I, 2005.

[FS93] C. Frougny and J. Sakarovitch. Synchronized rational relations of finite and infinite words. *Theoretical Computer Science*, 108(1):45–82, 1993.

[GGK91] P. Goralčík, A. Goralčíková, and V. Koubek. Alternation with a pebble. *Information Processing Letters*, 38(1):7–13, 1991.

[GH96] N. Globerman and D. Harel. Complexity results for two-way and multi-pebble automata and their logics. *Theoretical Computer Science*, 169(2):161–184, 1996.

[Gre70] S. Greibach. Full afls and nested iterated substitution. *Information and Control*, 16:7–35, 1970.

[Hay73] T. Hayashi. On the derivation trees of indexed grammars: an extension of the uvwxy-theorem. Technical report, RIMS Kyoto Univ. 9, 1973.

[HU79] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

[Imm88] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17(5):935–938, 1988.

[Kle56] S. Kleene. Representation of events in nerve nets and finte automata. In *Automata studies*, volume 34 of *Annals of Mathematics Studies*, pages 3–40. Princeton University Press, 1956.

[KN94] B. Khoussainov and A. Nerode. Automatic presentations of structures. In *Proceedings of the International Workshop on Logic and*

Computational Complexity (LCC 1994), volume 960 of *Lecture Notes in Computer Science*, pages 367–392, 1994.

[Kni64] J. D. Mc Knight. Kleene quotient theorem. *Pacific Journal of Mathematics*, pages 1343–1352, 1964.

[KNU02] T. Knapik, D. Niwinski, and P. Urzyczyn. Higher-order pushdown trees are easy. In *Proceedings of the 5th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS 2002)*, volume 2303 of *Lecture Notes in Computer Science*, pages 205–222. Springer, 2002.

[KNUW05] T. Knapik, D. Niwinski, P. Urzyczyn, and I. Walukiewicz. Unsafe grammars and panic automata. In *Proceedings of the 32th International Colloquium on Automata, Languages, and Programming (ICALP 2005)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1450–1461, 2005.

[KP99] T. Knapik and É. Payet. Synchronized product of linear bounded machines. In *Proceedings of the 12th International Symposium on Fundamentals of Computation Theory, (FCT 1999)*, volume 1684 of *Lecture Notes in Computer Science*, pages 362–373. Springer, 1999.

[Kur64] S. Kuroda. Classes of languages and linear-bounded automata. *Information and Control*, 7(2):207–223, 1964.

[LLS84] R. E. Ladner, R. J. Lipton, and L. J. Stockmeyer. Alternating pushdown and stack automata. *SIAM Journal on Computing*, 13(1):135–155, 1984.

[LS97] M. Latteux and D. Simplot. Context-sensitive string languages and recognizable picture languages. *Information and Computation*, 138(2):160–169, 1997.

[LS98] S. Lifsches and S. Shelah. Uniformization and skolem functions on infinite trees. *Journal of Symbolic Logic*, 63:103–127, 1998.

[Mar75] D. A. Martin. Borel determinacy. *Annal of Mathematics*, 102:363–371, 1975.

[Mas74] A. N. Maslov. The hierarchy of indexed languages of an arbitrary level. *Soviet Math. Dokl.*, 15:1170–1174, 1974.

[Mas76] A. N. Maslov. Multi-level stack automata. *Problems Information Transmission*, 12:38–43, 1976.

[Mey05] A. Meyer. *Graphes infinis de représentation finie*. PhD thesis, Univesité de Rennes 1, 2005.

[Min67] M. Minski. *Computations: finite and infinite machines*. Prentice Hall, 1967.

[Mor00] C. Morvan. On rational graphs. In *Proceedings of the 3rd International Conference on Foundations of Software Science and Computation*

Structures (FoSSaCS 2000), volume 1784 of *Lecture Notes in Computer Science*, pages 252–266. Springer, 2000.

[Mor01] C. Morvan. *Les graphes rationnels*. PhD thesis, Université de Rennes 1, France, 2001.

[Mos91] A. W. Mostowski. Games with forbidden positions. Technical report, University of Gdansk, 1991.

[MP04] A. Montanari and G. Puppis. Decidability of MSO theories of tree structures. In *Proceedings of the 24th International Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 3328 of *Lecture Notes in Computer Science*, pages 434–446. Springer, 2004.

[MR05] C. Morvan and C. Rispal. Families of automata characterizing context-sensitive languages. *Acta Informatica*, 41(4-5):293–314, 2005.

[MS85] D. E. Muller and P. E. Schupp. The theory of ends, pushdown automata, and second-order logic. *Theoretical Computer Science*, 37:51–75, 1985.

[MS97] A. Mateescu and A. Salomaa. Aspects of classical language theory. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 1, pages 175–251. Springer, 1997.

[MS01] C. Morvan and C. Stirling. Rational graphs trace context-sensitive languages. In *Proceedings of the 26th International Symposium on Mathematical Foundations of Computer Science (MFCS 2001)*, volume 2136 of *Lecture Notes in Computer Science*, pages 548–559. Springer, 2001.

[Ong06] C.-H. Ong. On model-checking trees generated by higher-order recursion schemes. In *Proceedings of the 21th IEEE Symposium on Logic in Computer Science (LICS 2006)*, pages 81–90. IEEE, 2006.

[Pay00] É. Payet. *Thue Specifications, Infinite Graphs and Synchronized Product*. PhD thesis, Université de la Réunion, 2000.

[Pen74] M. Penttonen. One-sided and two-sided context in formal grammars. *Information and Control*, 25(4):371–392, 1974.

[Pri00] Ch. Prieur. *Fonctions rationnelles de mots infinis et continuité*. PhD thesis, Université de Paris 7, 2000.

[Rab69] M. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

[Ris02] C. Rispal. The synchronized graphs trace the context-sensitive languages. In *Proceedings of the 4th International Workshop on Verification of Infinite-State Systems (INFINITY 2002)*, volume 68 of *Electronic Notes in Theoretical Computer Science*, 2002.

[Sak03] J. Sakarovitch. *Éléments de théorie des automates*. Éditions Vuibert, 2003.

[Sem84] A. Semenov. Decidability of monadic theories. In *Proceedings of the 11th International Symposium on Mathematical Foundations of Computer Science (MFCS 1984)*, volume 176 of *Lecture Notes in Computer Science*, pages 162–175. Springer, 1984.

[She59] J.C. Shepherdson. The reduction of two-way automata to one-way automata. *IBM Journal of Research and Development*, 3:198–200, 1959.

[She75] S. Shelah. The monadic theory of orders. *Annals of Mathematics*, pages 379–419, 1975.

[Sti00] C. Stirling. Decidability of bisimulation equivalence for pushdown processes. Technical Report EDI-INF-RR-0005, School of Informatics, University of Edinburgh, 2000.

[Tho97] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 389–455. Springer, 1997.

[Tho01] W. Thomas. A short introduction to infinite automata. In *Proceedings of the 5th International Conference on Developments in Language Theory (DLT 2001)*, volume 2295 of *Lecture Notes in Computer Science*, pages 130–144. Springer, 2001.

[Urv03] T. Urvoy. *Familles abstraites de graphes*. PhD thesis, Université de Rennes 1, France, June 2003.

[vM04] D. van Melkebeek. Time-space lower bounds for NP-complete problems. *Current Trends in Theoretical Computer Science*, pages 265–291, 2004.

[Wal96a] I. Walukiewicz. Monadic second order logic on tree-like structures. In *Proceedings of the 13th Annual Symposium on Theoretical Aspects of Computer Science (STACS 1996)*, volume 1046 of *Lecture Notes in Computer Science*, pages 401–414. Springer, 1996.

[Wal96b] I. Walukiewicz. Pushdown processes: Games and model checking. In *Proceedings of the 8th International Conference on Computer Aided Verification (CAV 1996)*, volume 1102 of *Lecture Notes in Computer Science*, pages 62–74. Springer, 1996.

[Wal02] I. Walukiewicz. Monadic second-order logic on tree-like structures. *Theoretical Computer Science*, 275(1–2):311–346, 2002.

[Wei04] P. Weil. Algebraic recognizability of languages. In *Proceedings of the 29th International Symposium on Mathematical Foundations of Computer Science (MFCS 2004)*, volume 3153 of *Lecture Notes in Computer Science*, pages 149–175. Springer, 2004.

[Wöh05] S. Wöhrle. *Decision problems over infinite graphs: Higher-order pushdown systems and synchronized products.* PhD thesis, RWTH Aachen, 2005.

Résumé

Cette thèse s'inscrit dans l'étude des graphes infinis de présentation finie. Nous nous intéressons à la fois à leurs propriétés logiques et aux langages qui leur sont associés. Nous nous concentrons sur l' étude des graphes infinis associés aux automates à pile d'ordre supérieur.

Notre première contribution est la définition d'une notion de rationalité pour les piles d'ordre supérieur. Nous montrons que cette notion partage de nombreuses propriétés de la rationalité sur les mots : clôture par complémentaire, accepteurs déterministes et complets, et caractérisation en logique du second ordre monadique. Nous établissons un lien étroit entre les automates à pile d'ordre supérieur et les ensembles rationnels de piles d'ordre supérieur.

Notre seconde contribution est l'étude structurelle des graphes associés ces automates. Nous en donnons différentes caractérisations qui montrent la robustesse de ces familles de graphes infinis.

Abstract

This thesis contributes to the study of families of finitely presented infinite graphs. We investigate their logical properties as well as the languages which are associated to them. We focus our attention on the infinite graphs associated to higher-order pushdown automata.

Our first contribution is the definition of a notion of regularity for higher-order pushdown stacks. We show that this notion shares similar properties with the notion of regularity on words: closure under complementation, deterministic and complete finite state acceptors and characterization by monadic second-order logic. We show a tight link between higher-order pushdown automata and the regular sets of higher-order pushdown stacks.

Our second contribution is the study of the infinite graphs associated to higher-order pushdown automata. We give various equivalent characterizations of these graphs and therefore establish the robustness of these families of infinite graphs.